UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

JOÃO CARLOS SILVA NOBRE

CONSIDERAÇÕES SOBRE ESTRUTURAÇÃO DE DOCUMENTOS PARA UTILIZAÇÃO DA XML NO COMÉRCIO ELETRÔNICO COM OBJETIVO DE FACILITAR A BUSCA DE INFORMAÇÕES

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação

Prof. Dr. Rogério Cid Bastos

Florianópolis, Fevereiro/2001

CONSIDERAÇÕES SOBRE ESTRUTURAÇÃO DE DOCUMENTOS PARA UTILIZAÇÃO DA XML NO COMÉRCIO ELETRÔNICO COM OBJETIVO DE FACILITAR A BUSCA DE INFORMAÇÕES

JOÃO CARLOS SILVA NOBRE

Esta Dissertação foi julgada adequada para obtenção do título de Mestre em Ciência da Computação na Área de Concentração Sistemas de Conhecimento e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da computação.

Prof. Dr. Fernando A. Ostuni Gauthier

Banca Examinadora

Prof. Dr. Rogério Cid Bastos (Orientador)

Prof. Dr. João B. M. Alves

Prof. Dr. Luiz F. J. Maia

“A mente que se abre a uma nova idéia jamais
voltará ao seu tamanho natural”
Albert Einstein

À minha esposa Maria Angélica pelo apoio solidário a cada dia desta jornada.

Agradeço a todos que contribuiram com a realização deste: Tribunal de Justiça do Estado de Rondônia, Fundação RIOMAR, Univesidade Federal de Santa Catarina, Rogério Cid Bastos, Maria Angélica Nobre, Christian Gravatá, Rodrigo Valério, meus pais…

Sumário

Resumo 1

Abstract 2

1. Introdução 3

1.1. Importância do trabalho 4

1.2. Objetivos 4
1.2.1. Geral 4
1.2.2. Específico 4

1.3. Limitações 5

1.4. Estrutura do trabalho 5

2. Comércio eletrônico 7

2.1. Fatores restritivos e críticas à Internet 10
2.1.1. Privacidade 10
2.1.2. Mensurabilidade 10
2.1.3. Congestionamento 10
2.1.4. Inadequação da legislação 10
2.1.5. Pornografia 11

2.2. Fatores restritivos ao comércio eletrônico 11
2.2.1. Custos 11
2.2.2. Segurança 12
2.2.2.1. Criptografia 14
2.2.2.2. Firewall 16
2.2.2.3. Secure Electronic Transactions - SET 20
2.2.2.4. Secure Socket Layer - SSL 23
2.2.2.5. Assinaturas digitais 24
2.2.2.6. As formas de pagamento 27

2.3. A tradição dos varejistas 31

2.4. Novas tecnologias em comércio eletrônico 32

2.5. Aspectos importantes sobre a implantação de comércio eletrônico 33
2.5.1. Iniciativas para o desenvolvimento do comércio eletrônico 35

2.6. A proposta da CommerceNet 35
2.6.1. A especificação RosettaNet 36
2.6.1.1. RosettaNet EC Managing Boards 37

2.7. Arquitetura OMG 37
2.7.1. Módulo de Dados Semânticos 38
2.7.2. Módulo de Negociação 39
2.7.3. Módulo de Pagamento Eletrônico 39
2.7.4. Módulo de Gerenciamento de Serviços 40
2.7.5. Módulo de Controle de Contratos 40
2.7.6. Módulo IPR – *Intellectual Property Right* 41
2.7.7. Módulo *Object Browser* – componentes para *Desktops* 41
2.7.8. Módulo de Agências 42
2.7.9. Módulo de Catálogos 42

3. Tecnologias envolvidas na criação de catálogos eletrônicos 44

3.1. Hipertext Markup Language - HTML 44

3.1.1. Pontos importantes a respeito da HTML 45
3.1.2. O formato da HTML 45
3.1.3. Vantagens e limitações da HTML 46

3.2. *eXtensible Markup Language* - XML 47
3.2.1. Documentos válidos e bem formados 49
3.2.2. As vantagens da XML 50
3.2.3. Elaboração de um documento XML 52
3.2.3.1. *Tags* de início e término 52
3.2.3.2. Designações de atributo 53
3.2.3.3. Referência a entidades 54
3.2.3.4. Comentários 54
3.2.3.5. Instruções de processamento 55
3.2.3.6. Seções CDATA 55
3.2.3.7. Declarações do tipo de documento 55
3.2.3.8. Um documento XML completo 56
3.2.3.9. Novas formas de pesquisa com XML 57
3.2.4. Pontos chave 59
3.2.5. Busca Convencional X Busca Otimizada 60

3.3. *Document Type Definition* - DTD 61
3.3.1. Definição DTD 61
3.3.2. Utilização da DTD em um documento XML 61
3.3.3. Detalhes para a construção de DTDs 62
3.3.4. Validação de um documento XML com o DTD 64

3.4. *eXtensible Style Language* - XSL 64
3.4.1. Princípios do modelo XSL 66
3.4.2. A arquitetura XSL 66

3.5. *Resource Description Framework* - RDF 68
3.5.1. O modelo de dados RDF 69
3.5.2. Exemplo da utilização de RDF 70
3.5.3. Vantagens do RDF 71

3.6. *Document Object Model* - DOM 71
3.6.1. O Modelo DOM 72
3.6.2. Vantagens do DOM 73

3.7. *Simple* API for XML - SAX 73
3.7.1. O funcionamento da SAX 74
3.7.2. Eventos SAX 75
3.7.3. Vantagens e Desvantagens da SAX 75

4. Catálogos Eletrônicos 77

4.1. Interatividade 77

4.2. Atualização dinâmica 77

4.3. Hipertextualidade 78

4.4. Presença global 78

4.5. Características funcionais dos catálogos eletrônicos 78

4.6. Tipos de catálogos eletrônicos 79

4.7. Catálogos e as novas formas de busca 82

4.8. Catálogos customizados 83

4.9. Catálogos interoperáveis 84

4.10. Modelo empresarial básico 88
4.11. Benefícios para os envolvidos na adoção dos catálogos 89
4.12. Uma visão prática dos catálogos 90
4.12.1. Publicação de um catálogo XML em um site da Web 90
4.13. O catálogo como uma ferramenta de vendas 93
4.14. XML Query Language - XQL 94
5. Java 97
5.1. Orientada a objetos 97
5.2. Interpretada 98
5.3. Portabilidade 99
5.4. Dinâmica e distribuída 100
5.5. Segura 101
5.6. *Multithreaded* 103
5.7. Servlets 104
5.7.1. Estrutura básica de um servlet 104
5.7.2. Vantagens dos servlets 106
5.8. *Commom gateway Interface* - CGI 107
6. Desenvolvimento da aplicação de busca em documento XML 108
6.1. Elaboração do catálogo 108
6.2. Criação do servlet para aplicação de busca na internet 111
7. Conclusões e considerações finais 116
7.1. Trabalhos futuros 117
8. Referências bibliográficas 119
Anexos 122
Anexo A – DTD 123
Anexo B – Folha de estilo XSL. 124
Anexo C – Catálogo XML 125
Anexo D - Código Fonte da Aplicação Busca Fácil 127

Índice de figuras

Figura 1 -Esquema geral de comunicação 15
Figura 2 - Uma Firewall 17
Figura 3 - Componentes de uma Firewall 18
Figura 4 - Assinatura digital baseada em criptografia assimétrica 26
Figura 5 - Message digest 27
Figura 6 - Principais Módulos 38
Figura 7 - Validação de um documento XML 64
Figura 8 - Geração de várias apresentações a partir de um mesmo documento XML 65
Figura 9 - Características mais importantes para um catálogo eletrônico 81
Figura 10 - Comparação da importância de navegação X pesquisa. 82
Figura 11 - Interação dos catálogos virtuais criados em XML 88
Figura 12 - Conversão de XML em HTML no servidor 91
Figura 13 - Conversão de XML em HTML no Cliente 92
Figura 14 -Representação do catálogo num navegador XML da Web 93
Figura 15 - Canal de comunicação ("push") com o cliente 94
Figura 16 - Site A 110
Figura 17 - Site B 110
Figura 18 – Interface da Aplicação Busca Fácil 112
Figura 19 – Resultado da Busca 113

Índice de tabelas

Tabela 1 - Exemplos de Tags em XML 53
Tabela 2 - Diferenças entre a busca convencional e a otimizada 60
Tabela 3 - Algumas convenções utilizadas na DTD 63
Tabela 4 – Relação compilador x código fonte da linguagem tradicional 98
Tabela 5 - Relação compilador x código fonte da linguagem tradicional Java 99

Índice de quadros

Quadro 1- Declaração de um resource ____ *70*
Quadro 2 - Exemplo de RDF ____ *70*
Quadro 3 – Instância da Classe Buscador ____ *113*
Quadro 4 - Determinação do menor preço para o produto ____ *114*
Quadro 5 – Criação da Instância da Classe SAXParser ____ *115*

Resumo

Com o desenvolvimento do comércio eletrônico, novas necessidades foram surgindo, como disponibilizar os produtos de maneira simples e atrativa, possibilitando ao comprador virtual facilidade na pesquisa, e possibilidade de encontrar rapidamente o que deseja.

A estruturação de documentos com a XML, tratando-se de catálogos eletrônicos ou não, permite a criação de mecanismos de busca eficientes, pois permite o processamento inteligente de documentos, apresentação e pesquisa baseada em conteúdo, independência de plataformas de *hardware* e *software*, compartilhamento e reutilização da informação.

Baseado nas características da linguagem XML, tornou-se possível o desenvolvimento do mecanismo de busca denominado Busca Fácil, o qual se apóia em quatro classes, sendo duas delas estendidas das classes HandlerBase e HttpServlet.

O sistema Busca Fácil, levando em conta os parâmetros da pesquisa solicitada, realiza a busca nos catálogos eletônicos, em sites cadastrados, os quais obedecem estruturação definida na DTD, e retorna ao usuário a resposta esperada de forma direta e simples, dispensando a usual procura em pesquisas via Internet, despertando interesse pela rapidez e praticidade.

Abstract

With the development of the electronic trade, new needs have appeared – the salesmen have to show the products in a simple and attractive way, making possible an easy research by the "virtual buyer", in the way to find quickly what he wants. That is the function of the electronic catalogs – instrument used by the salesmen to offer his products.

The structuring of documents through XML, referring to electronic catalogs or not, allows the creation of efficient search mechanisms, because it provides the intelligent processing of documents, the presentation and researches based on content, independence of hardware and software platforms, the sharing and new uses of the information.

Based on the characteristics of the XML, it was possible the development of the search mechanism named Busca Fácil (Easy Search), which leans on four classes, two of these classes are extended of the classes Handlerbase and HttpServlet.

The system Busca Fácil (Easy Search) makes the search in eletronic catalogs of registered sites, which obey defined structuring in DTD, and it backs to the user the expected answer in a direct and simple way, based on the parameters of the requested research. The results showed are opposed to the usual Internet researches, and it is very interesting because the search become speed and practice.

1. Introdução

Antigamente, fazer negócio estava associado a um passeio: pessoas iam ao mercado regatear seus produtos.

Hoje, não se precisa de dinheiro, nem mercadoria física e muito menos ir ao mercado para se fazer negócios.

Durante vários anos, a WEB foi considerada apenas um novo meio de comunicação, sem uma clara definição de aplicações voltadas aos negócios. Se essa vocação para negócios não tivesse sido percebida e aceita pela comunidade de usuários, a Internet não teria alcançado crescimento tão veloz, transformando–se numa ferramenta à disposição de considerável mercado consumidor.

A informação por si só, sua manipulação e recuperação, está se tornando um dos bens mais importantes no mercado *on–line*, proporcionando negócios inovadores e surgimento de novas formas de trabalho.

Devido a essas formas de negociação, estão surgindo várias questões a serem resolvidas para que o comércio eletrônico possibilite um mercado global. Muitos padrões estão evoluindo e várias empresas estão realizando trabalhos na tentativa de estabelecer meios e métodos comuns de negociação, envolvendo formas de pagamento, segurança, divulgação da informação.

E vários aspectos necessitam ser resolvidos, antes que uma inovação possa ser alcançada no mercado. Por causa da rápida evolução neste setor de negócio, as soluções para os problemas de implantação e padronização ainda são vagos, porém, não há dúvidas de que novos desafios e oportunidades estão surgindo, para clientes, empresas e instituições financeiras.

Apoiadas nas novas tecnologias, as empresas vêem no comércio eletrônico uma nova maneira de atuação, reduzindo custos, detectando o perfil dos clientes e aumentando a agilidade e os limites organizacionais.

Tendo em vista a crescente importância do comércio eletrônico, neste trabalho, almeja-se apresentar aspectos ainda não resolvidos e padronizados, sustentação para um maior desenvolvimento desse tipo inovador de comércio.

1.1. Importância do trabalho

O estudo da *eXtensible Markup Language* - XML se justifica pelo seu potencial uso em duas grandes áreas: a primeira seria a possibilidade de uma linguagem universal para definição de documentos, com sintaxes específicas para os mais variados domínios do conhecimento (matemática, química, comércio, etc.); e a segunda, pelo seu uso como um transportador de documentos no emergente segmento de comércio eletrônico, atuando na integração de aplicações e nos sistemas B2B (*business-to-business*).

1.2. Objetivos

1.2.1. Geral

Descrever os principais fundamentos a respeito do comércio eletrônico, comentar os padrões de soluções para o comércio eletrônico e demonstrar algumas considerações sobre estruturação dos documentos XML, direcionados ao desenvolvimento de catálogos eletrônicos, um dos componentes essenciais para integração de empresas participantes de negociações via internet.

1.2.2. Específico

O processo de padronização é muito complexo e envolve uma série de interesses econômicos e comerciais, os quais chegam a inviabilizar tentativas de padronização. Assim sendo, o objetivo específico deste trabalho estará centrado em fazer considerações sobre a estruturação das informações de um documento, adotando a tecnologia XML/DTD como facilitadora da integração entre parceiros e possibilitadora do desenvolvimento de mecanismos de busca de informações, apresentando os documentos XML como instrumento promissor à visada padronização.

A estruturação das informações será construída baseada nas necessidades dos usuários em manipular catálogos, eletrônicos ou não, para obter informações sobre produtos ou serviços oferecidos por entidades comerciais.

A adoção e utilização dessa estrutura, pelos parceiros de negócios torna exequível o desenvolvimento de um protótipo de catálogo eletrônico com a tecnologia XML, bem como a criação de um mecanismo de busca de informações na WEB, o qual se utiliza da estrutura de criação dos documentos XML para localizar, de forma rápida e

precisa, as informações solicitadas pelos usuários, visto que a busca é feita em parte determinada do documento XML.

Com a implementação de referido protótipo, pode ser observada a pontencialiadade do mecanismo de busca e, ainda, a estruturação da informação em um documento XML.

1.3. Limitações

A questão de segurança dos catálogos e transações via internet será abordada de forma teórica, pois descreverá as atuais técnicas usadas sem, contudo, tornar-se parte da aplicação desenvolvida.

A interface gráfica da aplicação não utilizará de estudos ergonômicos para o seu desenvolvimento.

1.4. Estrutura do trabalho

O presente trabalho está estruturado em sete capítulos. No Capítulo 1, tem-se uma breve exposição dos temas a serem discutidos no decorrer deste.

O Capítulo 2 descreve-se a fundamentação teórica referente ao comércio eletrônico, permitindo entender as relações entre comércio e tecnologia.

O Capítulo 3 apresenta a revisão bibliográfica voltada às tecnologias existentes para a construção de catálogos eletrônicos, um das principais formas de divulgação de produtos e serviços na Internet, e ainda exemplos da utilização de cada tecnologia.

No Capítulo 4, abordagem sobre a fundamentação teórica referente aos catálogos, tais como tipo, formas. Trata, também, da publicação de catálogos eletrônicos na Web.

O Capítulo 5 discorre sobre a tecnologia necessária à construção de aplicações executáveis pelos servidores web. A Linguagem Java é o tema principal desse capítulo, contendo ainda informações sobre *servlets* e *CGI*.

O Capítulo 6 tem como objetivo gerar os subsídios para fomentar a discussão sobre a eficiência da tecnologia empregada para a realização do objetivo específico. Além disso, descreve toda a tecnologia utilizada para elaboração do Sistema Busca Fácil.

O último Capítulo traz as conclusões e as propostas de trabalhos futuros.

2. Comércio eletrônico

Não existe uma definição exata e única sobre o termo, dependendo do ponto de vista de cada observador. Pode-se citar algumas como:

"E-Commerce on the one hand includes Electronic Shopping, but also means cooperation between Companies and their partners. So the meaning could be understood as E-Bussines, which includes all kind of communication between subjects in economy." [AUSWEGER & FRISCH, 2000]

"the sharing of business information, maintaining business relationships, and conducting business transactions by means of telecommunications networks … include[ing] the sell-buy relationships and transactions between companies, as well as corporate processes that support the commerce within individual firms"[1]

Comércio eletrônico é uma das formas de transação de negócios na qual as partes interagem por meio eletrônico, ou seja, compram e vendem produtos, serviços e informações utilizando-se de recursos tecnológicos, tais como rede de computadores, telefonia móvel e fixa.

Como se pode observar, o comércio eletrônico tornou-se um poderoso mecanismo de negociação para as empresas apoiadas em uma infra-estrutura tecnológica, provendo o crescimento de mercado de forma mais competitiva.

Descreve-se, abaixo, alguns termos envolvidos no comércio eletrônico.

One–to–One Marketing: Os Sistemas de comércio eletrônico passam a incorporar regras de negócio voltadas para a determinação do perfil dos clientes e oferecimento de promoções e produtos complementares. Através das técnicas de "*One–to–One Marketing*", personaliza-se totalmente as sessões de consulta de clientes a sites de Comércio Eletrônico, maximizando as possibilidades de venda e oferecendo um tratamento individualizado.

Produção "*Build to Order*" e "*Mass Customization*": Os produtos oferecidos em sites de comércio eletrônico passam a ser produzidos seguindo exatamente a especificação do

[1] ZWASS, Vladimir *in* [DANIEL, WILSON & PEPPARD, 1999]

cliente. Através da utilização de regras de negócio voltadas para a configuração de produtos, os sistemas podem guiar o usuário durante todo o processo de configuração, possibilitando a criação de produtos totalmente personalizados.

Customer Care: "Cuidar" bem do cliente, antecipando–se com relação a suas necessidades, também é um dos desafios dos sistemas de comércio eletrônico. Por exemplo, pode–se utilizar regras de negócio para automatizar a condução do relacionamento com o cliente através da emissão inteligente de *e–mails*.

Integração da Cadeia de Fornecimento (*Supply Chain Integration*): A integração entre os elementos de toda a cadeia de fornecimento (cliente, sites de comércio eletrônico, fornecedores, terceiros) passa a ser muito maior em relação aos métodos tradicionais. Sistemas que integram toda a cadeia de fornecimento consistem em uma nova filosofia de negócios.

Fabricação "*Just in time*": Os produtos passam a ser produzidos somente em função de pedidos específicos. Com isso, consegue–se uma diminuição nos níveis de estoque. Nas indústrias, a tendência é que sistemas de comércio eletrônico sejam conectados a softwares de ERP (Enterprise Resource Planning) no intuito de viabilizar a fabricação "just–in–time" em função dos pedidos oriundos da Internet.

Atualmente, o comércio eletrônico emerge no mercado e, como conseqüência, surge uma grande oportunidade de novos negócios num ambiente globalizado. Companhias estão mudando drasticamente para reduzir custos operacionais, melhorar o atendimento ao consumidor e estabelecer sua presença pelo mundo inteiro, independente da localização atual de seus escritórios. A convergência de comércio, rede de computadores e dinheiro são fundamentos básicos para a construção de um mercado eletrônico global. Hoje, computadores, soluções de rede e *software* simplificam e aumentam a comunicação entre parceiros em diferentes continentes e proporcionam práticas de negócio inovadoras.

Os varejistas acompanham de perto o chamado "marketing interativo", fascinados pela habilidade para se dirigir ao consumidor, colher e recuperar informações sobre os mesmos, incluindo, também, o atrativo da redução de custos através da eliminação da loja física e, conseqüentemente, dos custos associados a ela, fatores que lhes permitem triplicarem as suas margens de lucro [Cap Gemini, 1998].

Por outro lado, para os fabricantes, uma grande atração da rede é a possibilidade de comercializar mundialmente seus produtos. A facilidade de acesso ao consumidor pode fazer com que a distribuição direta se transforme na regra, deixando de ser a exceção.

Outro aspecto que chama a atenção refere-se aos impactos na maneira pela qual as empresas se comunicam com seus clientes. Atualmente, a forma como as empresas se relacionam com o mercado consumidor está mudando – a propaganda, a pesquisa de mercado e o *marketing* em geral estão saindo da divulgação em massa para a divulgação dirigida.

Nessa nova forma de negociação, surgem várias questões as quais necessitam ser resolvidas para que o comércio eletrônico e os sistemas de pagamento possibilitem um mercado globalizado. Com o grande crescimento da Internet, estão evoluindo os padrões para comércio eletrônico e formas eletrônicas de pagamento. É preciso, de alguma forma, definir quais são as preferências dos consumidores. E muitas questões surgirão e muitos caminhos serão suscitados antes de uma inovação ser alcançada no mercado. Por causa da rápida evolução neste setor de negócio, as soluções esperadas ainda se encontram em zonas vagas e obscuras. Oportunidades novas e desafios em negócio surgem para indivíduos, firmas e bancos.

O Comércio Eletrônico coloca à disposição um mercado novo para produtos existentes: isto significa que clientes podem ser servidos, via uma nova forma de distribuição de vendas. Fatos como o baixo custo da publicidade global e a possibilidade em alcançar um vasto número de clientes têm grande impacto na distribuição existente e no mecanismo de vendas. Além disso, as barreiras para pequenos negócios diminuem. Estas circunstâncias estão oferecendo uma boa oportunidade para idéias inovadoras.

Ao lado dos produtos existentes, novos serviços e produtos podem surgir, agregados ao ambiente *on–line*. Produtos baseados em informação, como bancos de dados *on–line* ou revistas eletrônicas, têm possibilidades de se tornarem novas categorias para a indústria. Também o setor de indústria chamado "edutainment", uma mistura de educação e entretenimento, começa a florescer nesse ambiente.

A administração pode melhorar suas decisões sobre metas estratégicas usando informação *on–line* para apoiar a decisão. A coordenação de processos de negócio, ou

workflows, pode ser aumentada e podem ser habilitadas operações mais rápidas ao longo da cadeia de suprimentos. Tais estruturas permitem companhias serem mais eficientes e flexíveis em suas operações internas, trabalharem mais de perto com seus fornecedores e serem mais rápidas no atendimento das necessidades de seus clientes.

2.1. Fatores restritivos e críticas à Internet

A Internet, com toda sua revolução e evolução, ainda possui pontos polêmicos, os quais geram críticas e fatores restritivos, dificultando o desenvolvimento desse novo meio de comunicação. Os principais são enumerados a seguir:

2.1.1. Privacidade

As comunicações e acessos feitos pela rede podem ser seguidos e espionados com facilidade, permitindo a leitura de correspondências e a identificação de *sites* visitados.

2.1.2. Mensurabilidade

Do ponto de vista empresarial, a medição da audiência da Internet tem gerado discussões e dúvidas, uma vez que ainda não há instrumentos e metodologias confiáveis para tal.

2.1.3. Congestionamento

Em países, como o Brasil, nos quais o sistema de telefonia estava, ou está, sem investimentos, a utilização da Internet tem sido bastante prejudicada em vários setores, pois necessitam da transmissão de dados, voz e imagens em movimento. Muitas aplicações se tornam impraticáveis pelas limitações de velocidades impostas pelo sistema telefônico.

2.1.4. Inadequação da legislação

Em vários países, a maioria das leis vigentes são anteriores ao surgimento da Internet e do comércio eletrônico, o que acarreta o aumento da insegurança para os usuários do sistema em geral [Ganança, 1998].

Paises como Suécia e Alemanha possuem uma legislação flexível e, portanto, atualizada, apoiando os contratos firmados entre os parceiros de negócio.

Recentemente, o Presidente dos Estados Unidos assinou o Ato *Electronic Signatures in Global and National Commerce* (E-Sign), o qual confere valor legal a contratos e ordens de compra, entre outros documentos, que são fechados *on-line*, considerado como um grande passo para o crescimento do comércio eletrônico em *business-to-business* e *business-to-consumer*.

Entre os países que já implementaram suas leis de assinatura digital estão a Malásia, Coréia do Norte e Singapura.

Na União Européia, o direcionamento é para que todos os países da comunidade implementem a assinatura digital.

No Japão, onde os selos corporativos são mais utilizados do que as assinaturas, o governo deve submeter uma lei ao parlamento esperando dar total respaldo legal às assinaturas digitais.

2.1.5. Pornografia

A pornografia que hoje circula pela rede, além de gerar grande volume de dinheiro, tem feito surgir reações conservadoras e tentativas de censura à Internet [Franco, 1999].

2.2. Fatores restritivos ao comércio eletrônico

O comércio eletrônico ultrapassará o tradicional após alguns problemas serem resolvidos, tais como: popularização dos computadores, facilidade de uso, maior aceitação e confiança nas transações entre as empresas e consumidores participantes da negociação.

2.2.1. Custos

Apresenta-se como uma das barreiras: além de pagar pela compra, o consumidor paga para comprar. Uma estratégia para fortalecer o varejo *on–line* está na idéia de se formar um pool entre os varejistas que oferecem seus produtos pela rede em associação com alguns provedores a fim de liberar o acesso gratuito aos consumidores que fazem compras através da Internet. No mundo real, o consumidor não paga para ficar dando voltas no *shopping*, entrando e saindo de lojas e escolhendo mercadorias. Na Internet, o usuário paga ao provedor de acesso pelo tempo de conexão e para a companhia

telefônica pelo consumo da linha. Seria tecnicamente possível, por exemplo, grandes *magazines*, através de acordos com os provedores, hospedarem seus *sites* com um procedimento o qual monitorasse automaticamente a conexão, liberando o tempo de acesso ao usuário sempre que ele esteja visitando seu *site*. Ainda assim, porém, o consumidor estaria pagando os impulsos da companhia telefônica. Para resolver este problema, só mesmo se os varejistas *on–line* funcionassem também como provedores de acesso através de linhas gratuitas. Nesse caso, admite-se que o acesso deva ser restrito apenas ao *site* da própria loja.

No Brasil, a internet grátis teve uma explosão em 1999. E, após o início do segundo semestre do ano 2000, algumas operadoras fecharam suas portas, pois a relação custo versus benefícios não se mostrou satisfatória. O mesmo aconteceu em países como os Estados Unidos, onde a internet tem maior número de usuários.

2.2.2. Segurança

Trata-se de um processo inevitável no futuro da Internet: a sua parte comercial acabará por se transformar num conjunto de lojas virtuais, onde tudo poderá ser adquirido e pago em segurança. O problema, por agora, é garantir essa segurança.

A Internet ainda é insegura de acordo com [GOMES, 2000] "A Internet nasceu em 1969. Após sua conclusão e estabilização, pesquisadores confrontaram-se com um fato: a Internet não é segura e pode ser facilmente fraudada. Atualmente escritores, para amenizar esse fato, ressaltam que a tecnologia de segurança empregada naquela época era muito primitiva. Há um pequeno engano. Hoje a tecnologia de segurança de rede é um pouco mais complexa e a Internet continua sendo fraudada."

De acordo com Schneier[2] está cada dia mais difícil proteger as informações no ciberespaço, e as empresas deveriam estar mais atentas e cientes da vulnerabilidade que decorre do desenvolvimento das formas de comunicação

Uma preocupação muito comum entre a maioria dos participantes da negociação eletrônica é relativa à segurança, reduzindo ou dificultando a comercialização entre clientes, parceiros. A atual ausência de segurança no pagamento *on–line* está impedindo

[2] *Secrets and Lies: Digital Security in a Networked World* (*Segredos e Mentiras: Segurança Digital num Mundo Conectado*), Schneier

a grandes segmentos da população efetuarem suas compras. Entretanto, a barreira está sendo superada e comerciantes *on–line* não deixam que as preocupações com a segurança no pagamento sejam obstáculo aos seus negócios na WEB, em um primeiro momento.

Com a popularização das redes de computadores, da Internet e do comércio eletrônico, todo sistema de computador se tornou um alvo em potencial para intrusos. O problema é: não há como saber os motivos que levam o intruso a agir e nem quando ele pode atacar. Contudo, é importante tentar se prevenir utilizando mecanismos de proteção, impedindo, ou pelo menos dificultando, o acesso por pessoas não autorizadas.

Associados à problemática da segurança, existem certos conceitos básicos que permitem normalmente definir o nível de segurança de um sistema. Esses conceitos são dados abaixo:

Integridade dos dados: A integridade dos dados permite a detecção de modificações não autorizadas nos dados. Vulgarmente, a integridade dos dados permite identificar se os dados foram modificados ou corrompidos durante a transmissão. Esta modificação pode ser resultado de um ataque ou de um erro na transmissão. A integridade dos dados pode ser conseguida através da implementação de uma função *one-way hash*.

Confidencialidade: É o processo utilizado para proteger informações secretas, de forma a não serem reveladas por pessoas não autorizadas. Os dados secretos devem ser protegidos quando são guardados ou transmitidos pela rede. Especificamente, essa proteção recorre ao uso da criptografia. A tarefa da implementação da criptografia também requer uma distribuição segura das chaves de criptação para o remetente e para o receptor dos dados cifrados.

Identificação: Os usuários são identificados, perante uma aplicação, por uma identificação do usuário, ou *userid*.

Autenticação: É o processo usado para verificar a identidade reivindicada por um usuário ou programa. A autenticação pode ser feita através do uso de *passwords* por parte do usuário, ou através da troca de chaves, e poderá eventualmente envolver uma terceira entidade de confiança.

Controle de acessos: O controle de acessos concede ou recusa a permissão a um dado usuário para acessar um recurso, limitando os acessos para os usuários autorizados. O controle de acessos é frequentemente especificado pelo administrador do sistema ou pelo dono do recurso.

Autorização: É o processo de atribuir acessos permitidos para o usuário. A permissão de acessos inclui uma especificação, como permissão para ler, escrever, ou alterar um dado arquivo.

Não repudiamento: É a capacidade de provar tecnicamente a origem dos dados e sua distribuição, ou seja, demonstra-se que a transmissão ocorreu, de fato, entre o remetente e o receptor. Desta forma, impede o remetente negar o envio dos dados, ou o receptor negar a recepção dos dados, e também impede que as entidades envolvidas possam alterar o conteúdo de tal transmissão.

Rejeição de serviço: O ataque de rejeição de serviços refere-se a um ataque no qual o atacante toma posse, ou consome recursos, de forma que ninguém mais possa usá-lo. Exemplos desses ataques incluem um vírus que consome a memória do sistema; ou um ataque na Internet, no qual o *host* atacante toma posse do *host* legítimo.

A seguir são descritas algumas tecnologias utilizadas na tentativa de solucionar os problemas com a segurança.

2.2.2.1.Criptografia

Criptografia é o estudo da cifragem e decifragem. Vem da palavra grega *kryptos* que significa “escondida”, e *graphia* cujo significado é “esc*r*ever”.

Criptografar significa transformar uma mensagem em outra – "escondendo" a mensagem original – com a elaboração de um algoritmo com funções matemáticas e uma senha especial, chamada chave. Os métodos de criptografia têm sido divididos em duas categorias: as de cifras de substituição e as de cifras de transposição [Tanenbaum, 1996].

A criptografia consiste na ciência (e arte) da transformação de mensagens numa representação sem significado para qualquer pessoa, exceto para quem saiba qual o processo de reverter a transformação.

A chave consiste em uma *string* que pode ser alterada sempre que necessário. Desse modo, o algoritmo de criptografia pode ser conhecido.

Todos os atuais algoritmos seguros são conhecidos e usam, no seu funcionamento, uma chave. As chaves devem definir univocamente o criptograma.

Em geral, a situação considerada é a necessidade de transmitir uma mensagem, entre um emissor e um receptor. O processo de disfarçar a mensagem chama-se cifragem e a transforma num criptograma. O processo de recuperar a mensagem original a partir do criptograma denomina-se decifragem.

Por sua vez, a criptanálise é a ciência (e arte) de quebrar criptogramas, ou seja, descobrir como fazer a decifragem de um criptograma sem saber a partida, ou como ele foi cifrado.

Os algoritmos de criptografia, também denominados cifras, são as funções matemáticas responsáveis pela cifragem e decifragem, apresentando, em geral, dois componente, respectivamente, o algoritmo de cifragem e o algoritmo de decifragem.

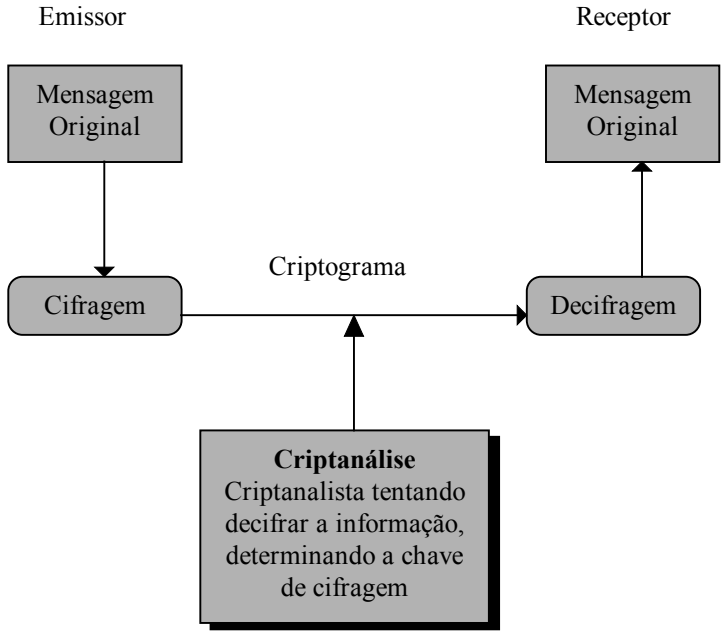


Figura 1 -Esquema geral de comunicação

O fator de trabalho para decodificar o método através de uma pesquisa no espaço da chave é exponencial em relação ao tamanho da chave. Por exemplo, com uma chave de tamanho de dois dígitos existem cem possibilidades, e em uma chave de tamanho de

seis dígitos existem um milhão de possibilidades; portanto, quanto maior for a chave, maior será o fator de trabalho com o qual o intruso terá de lidar.

Vantagens da Criptografia

- Proteger a informação armazenada em trânsito.
- Deter alterações de dados.
- Identificar pessoas.

Desvantagens da Criptografia

- Não há como impedir que um intruso apague todos os seus dados, estando eles criptografados ou não.
- Algum intruso pode alterar o programa para modificar a chave. Desse modo, o receptor não conseguirá decifrar com a sua chave.

2.2.2.2.Firewall

Outro item importante com relação à segurança necessária ao comércio eletrônico é a utilização da *Firewall*. Trata-se de um conjunto de *hardware* e *software* utilizado como alternativa para proteger computadores individuais na rede. Muitas organizações têm optado pela solução de proteger a sua rede interna de ataques externos com uma espécie de isolamento: as pessoas de "fora" não atacam a sua rede interna sem primeiro contatar suas premissas. Para criar esse isolamento é utilizada a *Firewall* que, do mesmo modo, também pode controlar o acesso da rede interna à Internet.

Firewall é uma barreira inteligente entre a rede local e a Internet, através da qual só passa tráfego autorizado. Este tráfego é examinado pela *Firewall* em tempo real e a seleção é feita de acordo com a regra "o que não foi expressamente permitido, é proibido". Pode ser usado tanto para bloquear o acesso a *sites* particulares na Internet, quanto para prevenir que certos usuários ou máquinas acessem certos servidores ou serviços, monitorando comunicações entre a rede interna e a externa. A *Firewall* também possibilita armazenar toda comunicação entre a rede interna e a externa e criptografar dados que são enviados pela rede.

Com a *Firewall* é possível estabelecer regras pelas quais o tráfego será selecionado. São selecionados os serviços da Internet, os endereços IP e as estações para as quais o tráfego será permitido ou negado. Este procedimento é conhecido como definir a política da *Firewall*.

Protocolo	Serviço/Porta	End Origem	End Dest	*in/out*	Negar/Aceitar
TCP	Telnet	204.17.195.0	199.232.92.0	*in*	Aceitar
TCP	FTP	200.13.142.1	199.232.89.0	*in*	Negar
TCP	Finger	201.11.100.0	199.232.90.0	*in*	Negar

(in/out = interface de entrada / saída)

Nesse exemplo, definiu-se protocolos e serviços que podem ser aceitos ou negados. Semelhantemente, há uma linha de permissões para pacotes.

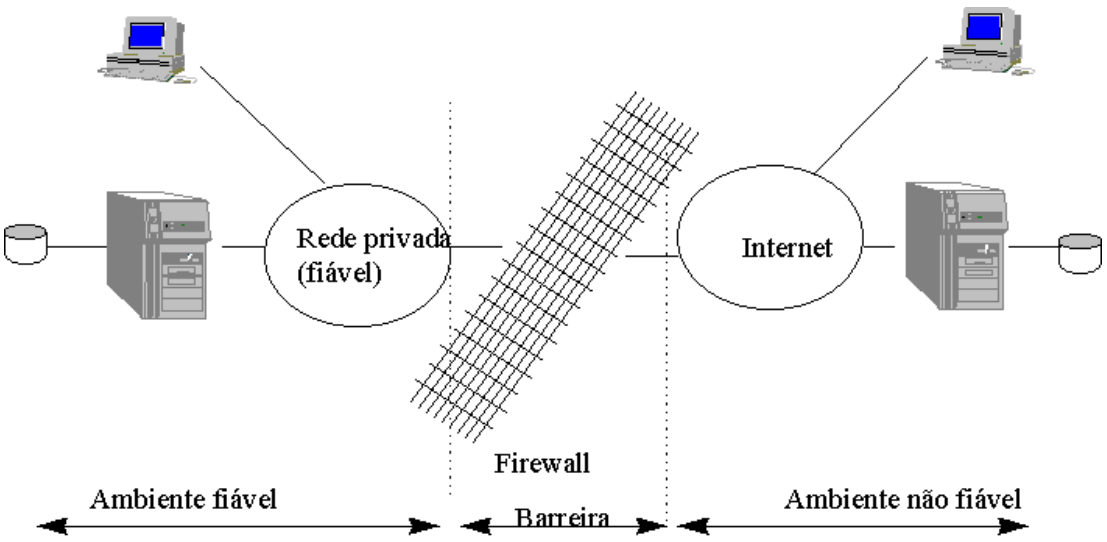


Figura 2 - Uma Firewall

Falhas que podem ocorrer com a *Firewall*

Todo sistema, por mais seguro que seja, está sujeito a falhas. As principais são:

- Se um usuário interno conectar na rede externa, ele criará uma porta de acesso à rede sem passar pela *Firewall*.
- Efetivo contra ataques externos, mas internamente a rede mantém-se vulnerável; e a maioria dos incidentes é causado por pessoal interno.

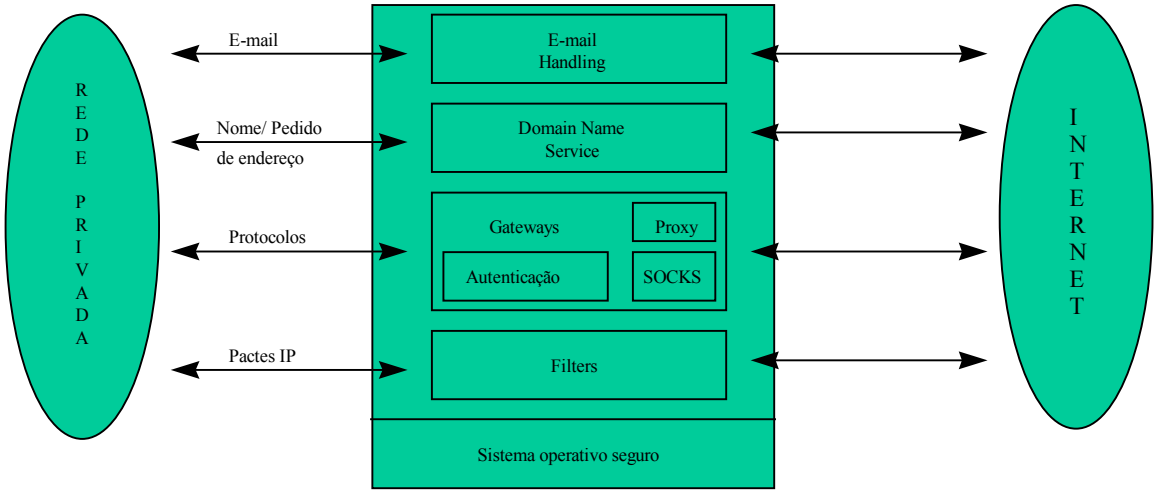
- *Bugs*: defeitos em programas de computador, na maioria das vezes muito prejudiciais ao sistema, como problemas de má configuração ou falha de equipamento, podem deixar a *Firewall* suspensa por algum tempo.
- Colisões da rede interna e externa podem evitar o acesso à *Firewall* por um instante, e justamente nesse instante um intruso poderá invadir a rede interna.

A representação de uma *firewall* consiste em vários componentes.

Os componentes de uma *firewall* encontram-se divididos em cinco grupos :

- Sistema operacional seguro.
- *Filters*.
- *Gateways.*
- *Domain name service.*
- *E-mail handling.*

Componentes de uma *Firewall*



Firewall

Figura 3 - Componentes de uma Firewall

Resumidamente, os componentes da *firewall* são descritos da seguinte forma:

O sistema operacional seguro protege não só o código da *firewall,* assim como outros arquivos contra os ataques de intrusos.

Relativamente aos *filters,* acrescenta-se o fato de executarem um conjunto de regras definidas pelo administrador da *firewall*; regras essas baseadas numa variedade de parâmetros como sendo o endereço de *IP*, os *ports* e ainda os tipos de aplicações. No que diz respeito ao endereço *IP*, ele próprio não é seguro. Como resultado disto, um dado *host* pode fazer *spoof* a um outro, mudando o endereço *IP source*.

Quanto aos *Gateways*, são responsáveis pela interceptação do tráfico enviado por uma aplicação, como exemplo *FTP*, e pela autenticação dos usuários no nível de aplicação *TCP*/*IP*. A função do *gateway* é, muitas vezes, concretizada através da implementação de um *proxy server*: quando um usuário da rede privada se liga a um *proxy server,* este autentica-o; depois da autenticação, o usuário liga-se ao servidor remoto na Internet. Deste modo, todas as comunicações efetuadas entre a Internet e a rede privada, serão recebidas, analisadas e admitidas pelos *proxy server*.

À medida em que um *proxy server* atua no nível aplicação, pode ser requerido um *proxy server* para cada tipo de aplicação. Conclui-se, então, que o *proxy server* autentica cada um dos usuários tanto do interior da rede privada como da Internet. Um forte sistema de autenticação é aqui exigido com vista a evitar aos usuários não autorizados sua entrada ou saída da rede privada.

No que diz respeito a *Socks*, também requer suporte *gateway* mediante a *firewall*. A principal diferença entre o *proxy server* e o *Socks server*, está no fato de o *proxy* requerer modificações no modo de acesso do usuário da Internet, sem a necessidade de se modificar o *software* do cliente; já o *Socks* requer modificações no *software* do cliente, mas não é efetuada nenhuma modificação nos procedimentos do usuário.

As *firewalls* incluem ainda um *Domain Name Service* e um *Mail Handling*. O primeiro isola o *name service* da rede privada da Internet. Como resultado, os endereços *IP* não são expostos aos usuários da Internet. A função desse serviço é, portanto, encontrar o endereço de uma máquina a partir do seu nome e vice-versa. A capacidade do segundo assegura que não haja nenhuma troca de *E-mail* processada através da *firewall*, entre a rede privada e a Internet.

2.2.2.3.Secure Electronic Transactions - SET

As tecnologias para transações comerciais, largamente utilizadas na WEB hoje, não suportam os diversos requisitos relativos à segurança do processo como um todo. O sigilo, a autenticidade, a integridade e a aceitação constituem os principais fundamentos para que os objetivos de segurança possam ser plenamente alcançados [DAHL, 1996]. Dentre os padrões propostos mais recentemente, destaca-se o SET (*Secure Electronic* T*ransactions*). Se por um lado, este padrão oferece soluções para todos os requisitos de segurança apresentados, por outro, a complexidade inerente ao processo de desenvolvimento de aplicações compatíveis com o SET e a ausência de frameworks mais objetivos têm dificultado a sua aceitação como um padrão de fato.

O SET foi desenvolvido por um consórcio de Empresas (Visa e Mastercard, além de GTE, IBM, Microsoft, Netscape, RSA, SAIC, Terisa e VeriSig), em 1996, tendo como premissa a criação de uma especificação aberta contendo vários métodos com a finalidade de trazer segurança às transações de pagamento via cartão de crédito sobre redes, como a Internet. A especificação SET é coordenada pelo *Secure Electronic Transaction Consortium*, (SETco).

A criação e o suporte do SET como um padrão aberto inclui em seus objetivos oferecer informações detalhadas sobre o protocolo, de forma que os vários sistemas possam trabalhar em conjunto, a nível global, utilizando, sempre que possível, os padrões já existentes. Outro objetivo é o de assegurar a compatibilidade com quaisquer combinações de hardware e software.

O SET visa conseguir a aceitação global através da facilidade de implementação; da integração com as aplicações clientes já existentes; da minimização das mudanças no relacionamento entre compradores, comerciantes e administradoras de cartão; e, por último, pela eficiência do protocolo para as instituições financeiras.

O SET apresenta sete grandes requisitos de negócio [LOEB, 1998]:

I. Oferecer confidencialidade sobre as informações relacionadas a pagamentos e pedidos de compra.

II. Assegurar a integridade das informações trafegadas.

III. Oferecer a autenticação de que o proprietário de cartão é um usuário legítimo de uma conta em uma administradora de cartões.

IV. Oferecer a autenticação de que o comerciante está habilitado para aceitar pagamentos da administradora de cartões em questão, através do seu relacionamento com uma instituição financeira.

V. Assegurar o uso das melhores práticas de segurança e técnicas de projeto de sistemas, a fim de proteger todas as partes envolvidas na transação de compra eletrônica.

VI. Criar um protocolo que não dependa de mecanismos de segurança no nível de transporte.

VII. Facilitar e encorajar a interoperabilidade entre provedores de *software* e serviços de rede.

O desenvolvimento do padrão SET trabalha com as seguintes entidades, envolvidas no esquema de segurança:

Autoridade Certificadora (*Certification Authority*): Emite os certificados digitais que garantem a autenticidade das partes envolvidas na transação. Possui a mesma função de um cartório do mundo real, ou seja, é responsável pelo processo de identificação e validação das partes. Para isso, define um processo para a prévia verificação dos dados de cada uma das partes e, após comprovar a autenticidade das informações, emite um documento digital (certificado) a ser utilizado durante a realização da transação eletrônica.

Cliente: realiza a requisição da compra (*Purchase Request*). Possui a carteira eletrônica (*wallet*) e acessa o servidor de comércio (loja eletrônica) através do *Browser*, selecionando, através de catálogos eletrônicos, os produtos que deseja obter. Em seguida, o cliente recebe um formulário de pedido contendo os itens escolhidos e o preço. O passo seguinte é selecionar o método de pagamento dentre os disponíveis em sua carteira eletrônica. O formulário e o método de pagamento são enviados para o servidor de comércio eletrônico.

Certificado Digital: é uma mensagem que contém informação sobre o seu detentor e que valida a sua existência, tal como um documento de Identidade. O certificado digital

do titular do cartão de pagamento equivale a uma representação eletrônica do mesmo, sendo, a sua autenticidade, assegurada pela inclusão de informações relativas à entidade emissora do cartão e à administradora do cartão. Com efeito, existe uma hierarquia na cadeia de certificação: a administradora do cartão certifica os Banco Emissores e de Apoio, os quais, por sua vez, emitem os certificados dos seus clientes, garantindo–se, assim, a legitimidade de todos os certificados. Quando da realização de uma compra, só em relação ao Banco será realizada a correspondência entre o número do certificado e os dados do cartão de pagamento (número e data de expiração), que não circularão pela Internet. Os certificados digitais dos comerciantes garantem igualmente a sua autentificação, haja vista incluírem informação sobre a entidade financeira que os apóia, bem como sobre a administradora do cartão. Pela exibição do certificado digital, o titular do cartão verifica a identidade do comerciante e a autenticidade da sua relação com uma instituição bancária, e vice–versa. Pelo número atribuído ao certificado digital do comprador, será possível ao Banco do comerciante identificar os dados relativos ao cartão a utilizar, evitando–se, assim, a circulação dessa informação na Internet. O certificado digital reside no computador do comprador, mediante a instalação de um *software* especial (wallet), reponsável por enviá-lo no início de uma compra virtual e por codificar as suas mensagens subseqüentes. A gravação do certificado digital em um chip–card, possibilitando a sua mobilidade entre vários pontos de acesso, está sendo desenvolvida.

Carteira eletrônica (*Wallet*): local onde o usuário guarda seus métodos de pagamento, ou seja, os dados sobre os cartões de crédito, débito e seus correspondentes certificados digitais. Quando um pedido de compra for realizado, a carteira é automaticamente ativada, e o usuário escolhe qual método deseja utilizar para pagar pela operação. A carteira também possui um extrato com todas as operações realizadas.

Servidor de comércio eletrônico (*Merchant Server*): permite a publicação dos catálogos eletrônicos das lojas virtuais. Realiza a interface entre a carteira eletrônica e o gateway de pagamentos. Cada loja virtual possui um certificado digital que garante a autenticidade da loja para o cliente. Dentro da especificação do protocolo SET, o lojista fica com as informações de venda, não tendo acesso aos dados do cartão ou ao método de pagamento selecionado pelo cliente. O Comerciante pode optar por efetivar a autorização imediatamente, ou então armazená–la localmente para posterior submissão

(Capture Request). Assim como a loja, a instituição financeira não tem acesso às informações dos bens e serviços comprados. Ao receber as informações da carteira eletrônica, o servidor de comércio pede uma autorização de pagamento para o *gateway* de pagamentos. As informações de pagamento são enviadas para o *gateway* de pagamentos, juntamente com o certificado digital do cliente e da loja. O Servidor confirma a compra para o cliente, fornecendo as informações sobre o prazo de entrega e número do pedido.

Instituição Financeira (*legacy*): faz a validação dos dados do cliente e emite uma autorização de compra ao *gateway* de pagamentos. Uma vez realizada a validação dos dados pela instituição financeira, essa envia uma "autorização-confirmação" para o comerciante (*Authorization Response*), confirmando a compra para o proprietário do cartão (*Purchase Response*).

***Gateway* de pagamentos (*Payment Gateway*)**: seu principal papel é receber mensagens de pagamentos provenientes dos servidores de comércio. Em seguida, envia mensagens à instituição financeira responsável pela autorização e pelo processamento da operação, seja ela de crédito ou débito, com a respectiva validação de informações em um sistema de retaguarda, o qual pode ser a conta corrente do cliente de determinado banco ou sistemas coorporativos da empresa administradora de cartões de crédito. O relacionamento entre cada componente garante a segurança e confiabilidade da transação eletrônica pela Internet. Em resumo, o Gateway confirma a efetivação da compra para o Servidor de comércio eletrônico, garantindo ter o cliente fornecido informações verdadeiras e possuir crédito para realizar a compra.

2.2.2.4.Secure Socket Layer - SSL

SSL é um protocolo para garantir a segurança de quaisquer dados em trânsito na Internet, desde que ambos, o servidor e o cliente, apóiem o protocolo. Foi desenvolvido pela Netscape para servir como um protocolo de segurança com a finalidade de realizar comunicações entre *Web browsers* e servidores. O SSL providencia autenticação, confidencialidade e integridade dos dados, sendo planeado para autenticar o servidor e, opcionalmente, o cliente. O SSL usa como protocolo de transporte o TCP, o qual provê uma transmissão e recepção confiável dos dados. Uma vez que o SSL reside no nível socket, ele é independente das aplicações de mais alto nível. Como tal, o SSL pode

realizar serviços seguros para protocolos de alto nível, como por exemplo TELNET, FTP, e HTTP.

O SSL é composto por dois protocolos: o SSL *Handshake Protocol* e o SSL *Record Protocol.*

O primeiro – SSL *Handshake Protocol* – é usado para negociar os parâmetros de segurança na conexão SSL; o segundo – SSL *Record Protocol* – especifica o encapsulamento de todas as transmissões e recepções de dados. Ambos fazem parte das negociações entre o cliente e o servidor, permitindo ao emissor identificar qual o algoritmo de cifragem suportado.

2.2.2.5.Assinaturas digitais

As assinaturas digitais providenciam a prova de autenticidade e origem dos dados.

Da mesma forma que muitas operações bancárias e outras transações tornam-se legalmente válidas depois de serem assinados certos documentos, na Internet, surge vulgarmente a necessidade de assinar documentos usando computadores locais. Contudo, uma vez que os computadores só aceitam informação na forma digital, qualquer assinatura em questão deverá, também, ser digital.

Partindo-se do pressuposto que as assinaturas digitais teriam de apresentar as mesmas propriedades das assinadas em punho, as assinaturas digitais deveriam ser:

- únicas.
- facilmente autenticáveis.
- não repudiáveis.
- baratas e fáceis de gerar.

Considerando que as duas primeiras, ao contrário da terceira, permitem manter a privacidade da mensagem, existem três aproximações possíveis para resolver esse problema:

- Assinatura digital baseada em Chave Secreta

Essa aproximação requer a existência de uma autoridade central a qual sabe de tudo e em quem todos confiam. Cada entidade escolhe uma chave secreta, dada a conhecer à autoridade central. Desta forma, só autoridade central e a própria entidade têm conhecimento da sua chave secreta. Quando uma entidade quer enviar uma mensagem assinada digitalmente a outra, terá de cifrá-la, com a sua chave secreta, e enviá-la à autoridade central. A mensagem passará pela autoridade central a qual a decifrará com a chave secreta da entidade emissora. A esta mensagem será concatenada uma *estampilha,* gerada e decifrada apenas pela autoridade central. O resultado será cifrado com a chave secreta da entidade receptora, e enviado. Desta forma, o receptor pode comprovar a recepção de qualquer mensagem através da *estampilha* recebida (só a autoridade central consegue produzir uma).

- Assinatura digital baseada em Chave Pública

Um dos problemas da aproximação usando chave secreta é a necessidade de todos em concordar com a autoridade central, a qual terá acesso ao conteúdo das mensagens. Os candidatos mais lógicos para serem esta autoridade central seriam o Governo ou os bancos. Contudo, tais organizações não inspiram confiança a todos os cidadãos. Devido a esse problema, surge a assinatura digital baseada em chave pública, a qual não requer qualquer autoridade central.

Essa aproximação pede que a entidade emissora saiba a chave pública da entidade receptora, e vice-versa. Tais chaves, em princípio, são diferentes. Assim, quando a entidade emissora quer enviar uma mensagem assinada digitalmente a outra entidade, terá de cifrar a mensagem com a sua chave privada e, em seguida, cifrar o resultado com a chave pública da entidade receptora. Por sua vez, a entidade receptora ao receber a mensagem terá de decifrá-la, primeiro, com a sua chave privada e, em seguida, decifrar o resultado com a chave pública da entidade emissora.

O receptor pode comprovar a recepção de qualquer mensagem através do criptograma resultante da decifragem com a sua chave privada. Note-se: ele consegue decifrá-lo, só que nunca conseguiria produzi-lo, uma vez ser desconhecedor da chave privada do emissor.

Esse método de assinatura digital tem todas as vantagens dos algoritmos de chave pública nomeadamente a sua impossibilidade de decifragem por outros, pelo menos em tempo útil.

Como exemplo de esquema de assinatura digital desse tipo cita-se o DSS (*Digital Signature Standard*).

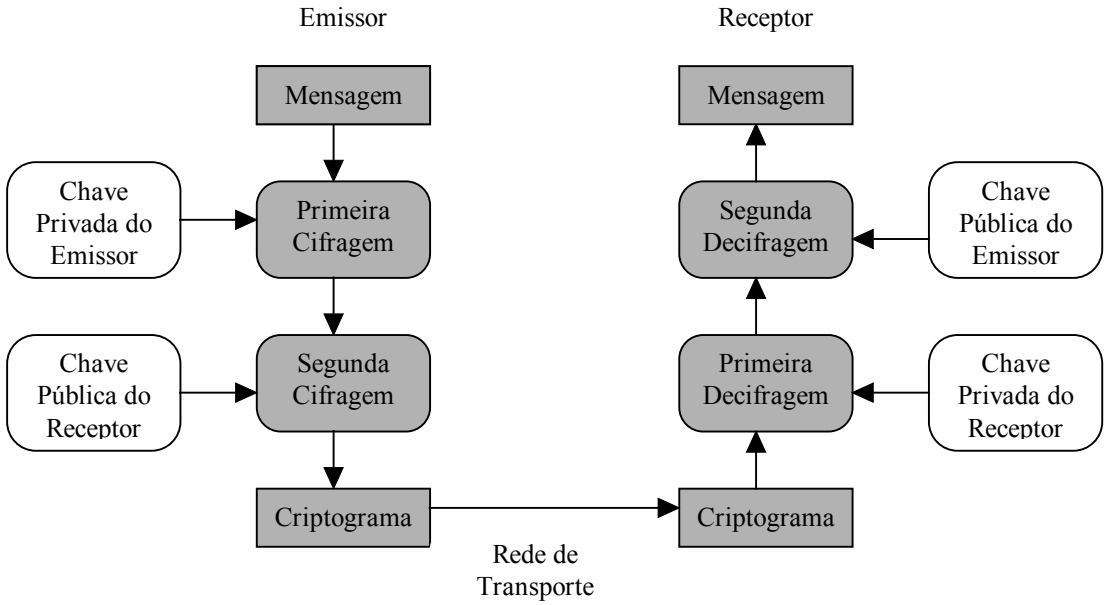


Figura 4 - Assinatura digital baseada em criptografia assimétrica

- Assinatura digital baseada em funções de sentido único

Uma das críticas às situações apresentadas anteriormente refere-se ao fato de unirem duas funções distintas: autenticação e privacidade. Muitas vezes, é necessária a autenticação, e não existe qualquer interesse de privacidade. Uma vez que a cifragem de uma mensagem com criptografia de chaves públicas é normalmente lenta, é freqüentemente desejável enviar uma mensagem assinada digitalmente sem preocupação de ser lida por outros. Desta forma, não se torna necessário cifrar toda a mensagem.

Tal esquema refere-se às funções de sentido único (*one-way hash funtions*) e tem como base a cifragem de uma parte, arbitrariamente longa, da mensagem, obtendo como resultado o chamado ***message-digest***.

Desta forma, a entidade emissora terá que gerar o *message-digest* e cifrá-lo (assiná-lo) com a sua chave privada. Após o que, poderá enviar a mensagem (não cifrada) concatenada com a sua *assinatura*. A entidade receptora decifrará a *assinatura* com a chave pública da entidade emissora (previamente publicada) e verificará se o *message-digest* é o esperado. Como pode ser facilmente percebido, as entidades comunicantes devem assegurar-se conhecerem as verdadeiras chaves públicas umas das outras, e não quaisquer outras ilegalmente publicadas, a ponto de comprometer a segurança do sistema. Para garantir isso, isto é, para fazer a distribuição de chaves públicas de forma segura, usa-se o conceito de **certificado**, um objeto que contém a chave pública de uma dada entidade assinada digitalmente por uma entidade de confiança, conhecida por **autoridade certificadora** (**CA**).

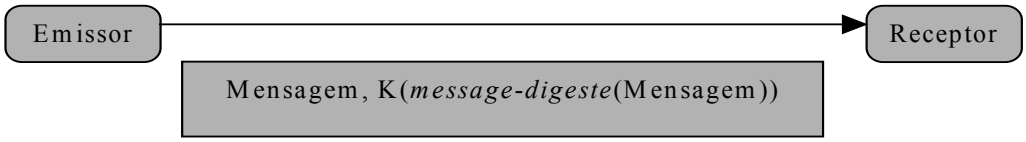


Figura 5 - Message digest

2.2.2.6.As formas de pagamento

O comércio eletrônico sem um serviço apropriado de sistemas de pagamento eletrônico não teria utilidade. A meta é a utilização de um meio comum para todo o processo, sem rompimento da cadeia empresarial presente nos processos de negócio.

As empresas são motivadas a utilizar sistemas de transferência eletrônica pela necessidade de entregar produtos e/ou executar serviços mais rapidamente, reduzindo custos e melhorando qualidade. Desde 1940, estão sendo realizadas pesquisas na área de sistemas de transferência eletrônica, porém, somente em 1968, ocorreu o primeiro marco importante, com a invenção do cartão de crédito. O segundo impacto importante aconteceu nos anos 1970, com sistemas chamados EFT (*electronic funds transfer*). “EFT é qualquer transferência de capitais iniciada através de um terminal eletrônico, instrumento telefônico, computador ou fita magnética" [Kalakota,1996].

O EFT usa novas tecnologias (como computadores, telecomunicações) para prover transações financeiras ou transferência de dinheiro. Os sistemas de pagamento têm evoluído, desde 1400, quando foram abertos os primeiros bancos na Itália,

intensificando-se, principalmente, de 1980 até nossos dias. Sua evolução completa pode ser encontrada em [Kalakota/Whinston, 1996].

Inovações de pagamento eletrônico, com características apropriadas, são mais indicadas no processo mercantil de comércio eletrônico do que os sistemas de pagamento tradicionais. Sistemas de pagamento eletrônico, ou sistemas de dinheiro vivo como são chamados, estão se tornando a parte central em negócios *on–line*. Os sistemas tradicionais de pagamento entre os participantes do processo mercantil – clientes, companhias, bancos – não são indicados, devido à demora do processo de pagamento ou o uso de vários mecanismos de pagamento entre os participantes do negócio.

Na situação atual dos pagamentos eletrônicos *on–line*, encontra-se, em muitas formas, remanescentes das idades medievais [Kalakota/Whinston, 1996]. Os comerciantes antigos enfrentaram vários problemas, como leis e alfândegas, relativos à impossibilidade da conversão de moedas. O meio apropriado para fazer troca ou compra de bens e serviços é a questão-chave, desafiadora no futuro. Certamente, novas formas de pagamento serão desenvolvidas, conforme a necessidade de velocidade e segurança impostas para as transações eletrônicas e os sistemas de pagamento tradicionais são os principais gargalos que dificultam, em parte por não serem adequados, as transações no ambiente de comércio eletrônico.

Os métodos de pagamento eletrônico (chamados de *cash systems*) para o cenário comercial podem ser divididos em mecanismos digitais baseados em moedas (*token–based*), cartões inteligentes (*Smart Cards*) e sistemas baseados em cartões de crédito [Kalakota/Whinston, 1996].

A mágica de criação de dinheiro eletrônico se deve à transformação de uma cadeia de caracteres, através do processo de criptografia, em valor em dinheiro, compreendido apenas pelo computador que o gerou, ou por leitores especiais. Essa característica faz o dinheiro eletrônico mais seguro, tornando sua falsificação muito mais difícil, em comparação às formas convencionais de pagamento (dinheiro, cheque, etc.).

A utilização do dinheiro virtual

O dinheiro virtual não vai ser aceito imediatamente. Qualquer sistema necessita ser seguro e rápido.

Para melhor se entender a utilização do dinheiro digital, é necessário abordar a utilização de um dos principais tipos do mercado:

E–cash

Segurança com o *E-cash*

O E-cash engloba os melhores atributos dos vários sistemas e com poucas técnicas de criptografia necessárias para utilizar qualquer um dos outros. O funcionamento do *E-cash* processa-se como um caixa multibanco e, os pagamentos, como se fossem ao balcão, permanecendo, o comprador, anônimo. Desenhado para permitir a efetivação, a partir de um computador pessoal, de pagamentos seguros para qualquer estação de trabalho através da Internet ou utilizando o *e-mail*, o E-cash oferece a privacidade do dinheiro em papel e uma alta segurança na sua utilização.

Conjunto de operações que suportam o *Ecash*

Quando é efetuado um levantamento, o *software* cliente do *E-cash* calcula a quantidade e o tipo de moedas envolvidas. A seguir, atribui um número de série aleatório e a informação é enviada para o banco digital na Internet, no qual cada usuário de *E-cash* tem uma conta aberta. Esse codifica os números com uma chave secreta – o dinheiro é autenticado – e debita na conta do cliente a quantidade de dinheiro pretendida. As moedas autenticadas são, então, enviadas para o usuário. Os números de série mais as assinaturas digitais (identificadores que determinam o valor de cada moeda) constituem moedas digitais e o seu valor é garantido pelo banco.

Quando o usuário pretende fazer uma compra em loja na Internet a qual aceita *E-cash*, o usuário envia as moedas necessárias para realizar o pagamento. Para tanto, a loja, após receber tais moedas, envia-as para o banco que verifica a sua validade. Se forem válidas, credita-as na conta do receptor do pagamento.

Facilidades oferecidas

Para além das facilidades retro mencionadas, ressalte-se que, um pagamento, no momento de ser realizado, a loja está *on-line* com o banco, e a verificação das moedas é efetuada imediatamente, porque os bens ou serviços comprados são imediatamente enviados para o cliente.

Com esta forma de pagamento, evita-se abrir conta na loja onde se efetua uma compra, ou de enviar informação sobre o cartão de crédito.

Observe-se também que, cada moeda só poder ser utilizada apenas uma vez, pelo que não será bem sucedida qualquer tentativa de usar cópias de moedas. Assim, quem as recebeu e pretende gastá-las terá de fazer um novo levantamento.

O dinheiro nunca é perdido: se as moedas forem apagadas acidentalmente do disco rígido, basta enviar os números (únicos) das séries perdidas e o montante será devolvido ao usuário.

E ainda, o fato de o banco poder identificar o recebedor permite minimizar as possibilidades de utilização de *E-cash* de forma ilícita.

NetCheque

O *NetCheque* é outra forma de pagamento eletrônico na Internet. Tal como os cheques de papel, os *NetCheque* são simples documentos (eletrônicos enviados por *e-mail*) que incluem o nome do pagador (o titular da conta), a identificação da instituição financeira, o número da conta bancária, o valor do cheque e o nome de quem o vai receber. Para sua autenticação, cada *NetCheque* possui uma assinatura digital, ou seja, um código criptografado.

Quando depositados, esses cheques autorizam a transferência da quantia mencionada, de uma conta para outra.

Vale ressaltar que todo o processo está protegido por um sistema de criptografia *Kerberos*.

A utilização de cartões de crédito

Por mais de três décadas, os cartões de crédito têm sido usados como forma de pagamento universal. A popularização deste meio de pagamento, especialmente nas camadas de maior poder aquisitivo, se deve, principalmente, à sua universalidade (independência de um padrão monetário) e sua facilidade/praticidade de uso.

Outrora, apresetavam-se questões sutis em relação ao uso do cartão de crédito, referem–se ao mecanismo das transações comerciais. Em operações realizadas diretamente entre o comerciante e o cliente, o lojista deve ser capaz de realizar uma

análise de crédito do comprador antes de efetuar a transação. O cliente, por sua vez, terá de se submeter às condições de financiamento do vendedor, no entanto, nem sempre o estabelecimento que oferece os melhores produtos e preços detém as melhores condições de financiamento. E, devido à natureza da transação, o setor do varejo corre o risco de prejuízos constantes com negócios mal sucedidos [Duarte, 1998].

Atualmente, no caso de uma transação efetuada através de cartão de crédito, o estabelecimento comercial tem a certeza do pagamento da mercadoria, realizado através de uma instituição financeira. O cliente teria, em princípio, a capacidade de negociar o financiamento diretamente com a instituição de sua preferência. Cada empresa cuidaria, assim, de seu negócio específico de maneira mais eficiente.

O cenário do comércio eletrônico se agrega a essa visão, em especial, o comércio eletrônico via Internet. Em um mundo sem fronteiras, é fato que a Internet se tornará o principal elemento integrador, redutor de distâncias e custos. Um ambiente como esse requer uma forma de pagamento tão universal ou mais, que a definida pelos cartões de crédito. No entanto, a Internet coloca à exposição, ainda mais, a principal falha operacional dos cartões de crédito: a segurança.

Na prática, em uma operação comercial pela rede, não há presença física do comprador e vendedor, aumentando a preocupação com o risco de fraudes e omissão da Nota Fiscal de compra.

2.3. A tradição dos varejistas

Os consumidores possuem grande interesse em utilizar a Internet como um canal de compras "diretamente da fábrica". Para os consumidores preocupados apenas com o preço, seria vantajoso comprar diretamente do fabricante, evitando os intermediários. Porém, surgiram algumas questões: os fabricantes que vendem através da Internet podem ou não rebaixar seus preços? Além disso, os consumidores da Internet estão muito interessados no auxílio ao cliente e na legitimidade do comerciante que se apresenta *on–line*. Os varejistas têm tradição de oferecer bom atendimento ao cliente e, se a souberem associar à conveniência da Internet, as vendas *on–line* se fortalecerão.

2.4. Novas tecnologias em comércio eletrônico

A implementação de Sistemas de Comércio Eletrônico responsável por atender aos requisitos necessários para viabilizar os benefícios citados nos itens acima, pressupõe a utilização de novas tecnologias e recursos computacionais. Algumas das mais importantes são:

- Agentes e Regras de Negócio - As regras de negócio contidas nos sistemas de comércio eletrônico devem compreender a possibilidade de ser rapidamente modificadas, uma vez que novos produtos e promoções sempre estarão sendo incluídos dinamicamente. Para tal, é extremamente desejável que as regras de negócio estejam separadas da lógica da aplicação, facilitando a manutenção e atualização das mesmas. As Regras de Negócio poderão ser encapsuladas em agentes inteligentes, programados em Java, responsáveis por atividades como determinar o perfil do cliente, auxiliá–lo na configuração de produtos, sugerir promoções e produtos complementares, etc.

- Servidores WEB com conteúdo dinâmico - Em função do grande volume de informações a ser constantemente atualizado, um dos requisitos dos Sistemas de Comércio Eletrônico é permitir ao conteúdo das páginas HTML ser modificado dinamicamente. Dentre os Servidores WEB que suportam a construção de páginas com conteúdo dinâmico, destaca–se o "Domino", da Lotus, utilizado pela "*Commerce One*" em projetos de Sistemas de Comércio Eletrônico. Aplicações "Java *Server–Side*" também podem ser utilizadas para criar sites de conteúdo dinâmico.

- Modelo "*Thin client*" – Já que existe a tendência da ampliação dos Sistemas de Comércio Eletrônico de forma a ser acessados a partir de qualquer ponto e, levando–se em consideração as limitações de velocidade em conexões telefônicas, é interessante utilizar uma arquitetura baseada no conceito "*thin client*", no qual as informações e programas enviados ao *Browser* do usuário são mais compactas, diminuindo principalmente o tempo de espera para a carga da página, obviamente, dependendo do tipo da aplicação que está sendo executada.

2.5. Aspectos importantes sobre a implantação de comércio eletrônico

Se, por um lado, o comércio eletrônico apresenta grande potencial, por outro é possível dizer que as empresas interessadas em iniciar operações de comércio eletrônico devem fazê–lo com cautela, alimentando uma visão de longo prazo, buscando coerência e integração na sua estratégia empresarial, avaliando cuidadosamente as sinergias oferecidas pela rede ao seu negócio e as ameaças trazidas por essa nova modalidade de comercialização à sua estrutura de distribuição. Como exemplo, segue uma lista de questões a ser respondida pelo varejista antes de iniciar suas operações de comércio eletrônico [Morgado, 1998]:

- Como integrar o comércio eletrônico à estratégia geral da empresa?
- Quais os tipos de produtos a serem comercializados?
- Qual deve ser a definição de público alvo nos negócios eletrônicos?
- Como desenvolver abordagens de comunicação individualizadas?
- Que tipo de conteúdo e de informação deve ser veiculada?
- Como combater os atuais problemas trazidos pela baixa velocidade de transmissão?
- Como lidar com essa nova mídia, onde o cliente conta com um poderoso canal de comunicação no qual ele pode passar a ser um agente emissor?
- Como mensurar os resultados e qual deve ser a expectativa de retorno?
- Como lidar com a segurança nas transações?
- Como garantir a privacidade dos clientes?
- Como criar interações e diálogos eletrônicos fáceis e ao alcance da grande população?
- Como divulgar uma loja virtual?
- Como escolher os fornecedores de informática, criação e comunicação para a nova mídia?

- Quais os cuidados com a criação e *lay–out* das interfaces com o consumidor?

A nova geração de sistemas de comércio eletrônico deve ser dotada de uma arquitetura modular a qual possibilite a implementação de soluções escalonáveis, permitindo se iniciar com um tamanho reduzido a ser gradativamente expandido. Esse ambiente possibilita a implementação de algumas funções que podem agregar muito valor às soluções de comércio eletrônico:

a) Promoções

- Determinação automatizada de preços e descontos através de regras de negócios.
- Possibilidade de customização de programas de descontos de forma única para cada usuário.
- Integração com sistemas transacionais existentes.

b) Seleção Dinâmica de itens

- Determinação de requisitos através de perguntas estratégicas.
- Associação da necessidade do cliente aos produtos que melhor a satisfaçam.
- Recomendação automática de itens.

c) Controle da Oferta de Itens

- Priorização da Oferta de itens.
- Sugestão de itens alternativos, caso o item desejado pelo cliente não se encontre disponível.

d) Configuração

- Configuração automatizada de produtos e serviços.
- Seleção dinâmica de opções com base em regras de configuração.

2.5.1. Iniciativas para o desenvolvimento do comércio eletrônico

Várias iniciativas têm emergido com o objetivo de desenvolver o Comércio Eletrônico, visando definir a forma com a qual parceiros de negócios devem se comunicar. Entre elas, destacam–se "CommerceNet" e "RosettaNet".

2.6. A proposta da CommerceNet

CommerceNet é a primeira associação industrial que utiliza, promove e constrói soluções de comércio eletrônico para a Internet. Sua missão é facilitar a utilização do comércio eletrônico, tornando–o confiável e onipresente. CommerceNet e seus sócios, submeteram a proposta de um projeto Piloto ao Governo Federal Americano, com o propósito de demonstrar os benefícios econômicos e viabilidade técnica da utilização de catálogos, dentro do contexto de um sistema de busca eletrônico, para ser utilizado por várias agências do governo.

O Projeto Piloto apresentado enfatiza a reconhecida existência, tanto por parte da indústria como do governo, de um enorme custo e uma baixa eficiência, inerente ao processo de cotação de preços em catálogos de vários fornecedores, representando um alto custo, principalmente quando várias agências extraem a mesma informação de um mesmo fornecedor. Portanto, o processo de cotação se repete multiplicando os custos. O fornecedor também deve apresentar múltiplos orçamentos para o governo, através de uma grande quantidade de catálogos, a princípio, eletrônicos. Tal empreita é freqüentemente impossível para os fornecedores menores, sem recursos necessários para disponibilizar uma grande quantidade de catálogos *on–line*. Dessa forma, torna–se evidente e necessária a criação de um sistema a envolver o governo e os fornecedores para que troquem informação de uma forma mais eficiente. Contudo, é preciso uma implementação não limitadora da flexibilidade necessária a cada agência governamental, nem tão pouco prejudique ou forneça vantagem competitiva a qualquer fornecedor. A chave para resolver esse problema está na geração de uma solução paasível de tornar os catálogos reconhecidos tanto pelas agências do governo como pelos fornecedores. Devido à grande variedade de fornecedores envolvidos, existe a necessidade de padronizar várias representações independentes, ligadas a vários tipos de mercados. A solução deve englobar também os aspectos inerentes ao comércio eletrônico, como critérios de autenticação, segurança, pesquisa, entre outros. Esse

projeto Piloto seria construído usando uma arquitetura definida como "*CommerceNet's eCo framework*" usada para criar um modo padrão de representação da estrutura dos tipos de informação nos documentos, troca de mensagens e formatos de arquivo entre as aplicações.

A proposta se fundamenta na resolução dos problemas citados, criando um ambiente no qual todos possam competir livremente no mercado, e no qual cada organização poderá manter sua identidade. Mais especificamente, a intenção é demonstrar que, com esse projeto Piloto, é possível:

- Reduzir os custos do alto volume de processamento de transações de pequeno valor.
- Reduzir a incidência de comissões ilícitas e compras super faturadas.
- Reduzir o tempo que os usuários levam para encontrar os produtos desejados.
- Reduzir o tempo do ciclo de pedidos.
- Melhorar a qualidade da informação para entidades de compra governamentais.
- Simplificar o processo para que vendedores possam expor seus produtos para as várias agências do governo.

2.6.1. A especificação RosettaNet

A missão de RosettaNet é aproveitar o alcance global e penetrante da Internet, definindo e conduzindo a implementação de processos eletrônicos de negócio, abertos e comuns. Esses processos são projetados para padronizar as interfaces eletrônicas de negócios entre os parceiros empresariais da cadeia de suprimentos, resultando em benefícios mensuráveis. RosettaNet chama referidos processos de negócios comuns e abertos de diretrizes PIP (*Partner Interface Process*). Essas diretrizes PIP incluem especificações legíveis para humanos e máquinas, usadas para monitorar a conformidade das especificações RosettaNet na interação entre os parceiros do negócio eletrônico e as aplicações.

As organizações parceiras do negócio na cadeia de suprimento utilizarão essas diretrizes PIP para configurar seus processos de negócio de acordo com as outras organizações participantes. Destaque para as aplicações em rede, as quais colaboram

entre si para executar os processos de negócio entre as organizações parceiras: elas inter– operam, aderindo à arquitetura aberta e comum da RosettaNet.

O modelo de negócio da RosettaNet pretende habilitar os parceiros de negócio da cadeia de suprimentos a executar processos de negócio inter–operáveis, isto é, processos que podem ser manipulados por vários sistemas de diferentes empresas, desenvolvendo e distribuindo interfaces.

2.6.1.1.RosettaNet EC Managing Boards

É uma iniciativa da RosettaNet responsável em desenvolver uma interface para ser utilizada pelas empresas participantes do projeto. Uma vez definida a interface, os participantes se responsabilizam em implementá–la e divulgá–la entre seus parceiros de negócio. Pretende–se, com isso, definir uma forma única para a identificação dos produtos e serviços envolvidos na cadeia de suprimento, padronizando preços, data de envio, etc. Essa interface se baseia, principalmente, na padronização de catálogos contendo, inicialmente, a definição de 3600 itens de comum acordo entre os participantes. *RosettaNet EC Managing Boards* consiste em vinte e nove empresas globais pertencentes a uma cadeia de suprimentos na área de informática, envolvendo fabricantes e fornecedores de chips, distribuidores, fornecedores passivos, fornecedores de conectores – tais como 3Com, Cisco, Compaq, Hewlett–Packard, Intel, Microsoft, Netscape, SAP – e alguns clientes especiais – como *U.S. General Services Administration.*

2.7. Arquitetura OMG

O OMG (*Object Management Group*), um comitê com o objetivo de promover teoria e prática da terminologia de objetos para o desenvolvimento de sistemas distribuídos, estabeleceu, em julho de 1997, as tecnologias e serviços essenciais ao comércio eletrônico, demonstrando seus aspectos principais, visando fornecer uma arquitetura comum para aplicações orientadas a objetos e habilitando um ambiente de mercado aberto. É essencial o estabelecimento de uma arquitetura para identificar as relações e papéis entre os participantes do mercado. A especificação OMG define os principais componentes necessários para dar suporte aos participantes das transações de comércio eletrônico, definindo as interfaces e semânticas de serviços. A seguir serão

abordados seus principais conceitos. A arquitetura OMG é dividida em três grupos principais, conforme a figura 6:

1 – Serviços de comércio eletrônico de baixo nível, incluindo sistemas de pagamento, semântica e serviços de seleção/negociação.

2 – Gerenciamento de serviços comerciais, suporte a contratos, suporte a desktop.

3 – Módulos de infra–estrutura envolvendo catálogos, negociação e agências.

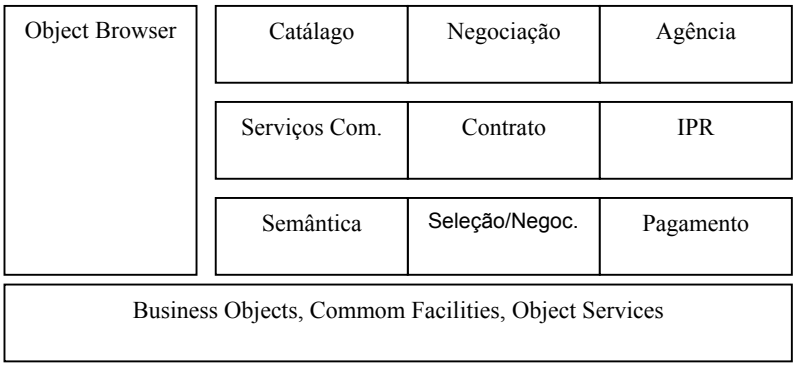


Figura 6 - Principais Módulos

2.7.1. Módulo de Dados Semânticos

Uma das exigência para o comércio eletrônico é a estrutura comum para facilidade na troca de dados, a qual diz respeito à própria informação em si (dados semânticos), descrevendo produtos, serviços, conteúdo e recursos. Atualmente, essa exigência é executada por applets Java que dão pouco apoio e não são suficientes para um serviço comercial.

Descrições de serviço comerciais exigem um modelo de objetos com nível conceitualmente alto, possíveis de serem passados rápido e eficientemente entre participantes em um mercado eletrônico (provedores de serviço, provedores de catálogos, corretores, consumidores), e facilmente reproduzidos, independentemente da fonte da informação.

Uma segunda exigência consiste na necessidade da construção de descrições dinâmicas de serviços e sua modificação subseqüente, ou até a extensão de tais descrições (possivelmente como resultado de negociação entre participantes de mercado).

Um dado semântico é considerado como uma informação, a qual descreve algum produto ou um objeto qualquer. Adicionalmente, um objeto de dados semântico pode incluir uma informação específica a uma aplicação distinta. Característica inerente a um objeto de dados semânticos (SDO) é a habilidade para descrever qualquer tipo de serviço a ser oferecido num mercado eletrônico. Considerando que um SDO é transferido e processado no computador local do usuário e por não ser possível prever quais elementos são necessários para descrever um serviço específico, o SDO deve ser suficientemente geral para permitir a definição de novos tipos, obrigando a inclusão de meta–informação (dados sobre os dados) dentro de um SDO.

2.7.2. Módulo de Negociação

Esse módulo tem a função de criar uma negociação virtual entre dois ou diversos participantes de uma relação comercial. Através de *software*, são criadas todas as condições necessárias para que haja uma negociação, incluindo acordo, aceitação, rejeição e pendência de uma transação entre os interessados.

2.7.3. Módulo de Pagamento Eletrônico

O avanço do Comércio Eletrônico, na Internet, criou novas exigências para pagamentos eletrônicos, como também diversificou as formas, como por exemplo, cartões de crédito, cartões de débito, dinheiro vivo eletrônico, micro pagamentos. Pagamentos eletrônicos envolvem muitos tipos diferentes de pagadores – bancos, consumidores, comerciantes – sendo preciso criar uma forma de troca eletrônica consistente e segura para atender às diversas necessidades do mercado. Protocolos para gerenciamento das relações entre os participantes do comércio eletrônico estão sendo desenvolvidos por diversas empresas, para satisfazer necessidades específicas, conforme já citado.

Essas interações entre comprador e vendedor, e qualquer outra parte envolvida, são necessárias para a troca eletrônica de valores de bens ou serviços. A troca de valores é controlada pelas exigências do negócio, expressas em um protocolo de pagamento e associadas com cada tipo de pagamento, os quais podem ser:

- Cartões de crédito.

- Cartões de débito.
- Dinheiro eletrônico.
- Micro pagamentos.

2.7.4. Módulo de Gerenciamento de Serviços

Os consumidores possuem uma precaução adicional com relação ao comércio eletrônico, pois comércio significa dinheiro. Infra–estruturas atuais não fornecem apoio necessário, gerando desconfiança nos serviços empresariais. Insurge, por isso, a busca de sistemas possibilitadores de uma comunicação segura entre as diferentes interfaces utilizadas pelo consumidor, pelo provedor, e por qualquer outra entidade envolvida, em uma transação de comércio eletrônico. Serviços podem ser considerados processos direcionais: de um lado está o provedor e do outro, o consumidor; não são trocas. Exemplo de serviços no mundo de comércio eletrônico incluem: pagamento eletrônico, o qual é feito por um cliente (o provedor) para um comerciante (o consumidor) através de um portal de pagamento (o terceiro envolvido, como um banco). Para cada tipo de participante (o consumidor, provedor ou terceiro), um objeto de serviço define a interface e o comportamento, fornecendo as interfaces:

Interface de consumidor: define o estado atual de um serviço, as operações disponíveis e a política aplicável a todos os participantes engajados no serviço.

Interface de provedor: define a interface que um provedor está declarando (a declaração das ações permitidas ao provedor no curso de entrega do serviço) e o comportamento a ser mantido por ele. Por exemplo, uma interface de provedor pode possuir uma operação a qual permita cancelar um compromisso durante uma fase particular da execução do serviço.

Interface do terceiro: define as operações e comportamentos de qualquer outra parte envolvida nos serviços.

2.7.5. Módulo de Controle de Contratos

O contrato abrange um plano de serviço, contendo os serviços possíveis de serem realizados entre os participantes, além da sua seqüência de execução e assinaturas do contrato. O plano de serviço pode ser elaborado em comum acordo entre os

participantes do contrato. O modelo de contrato também é constituído por uma coleção de assinaturas dos envolvidos (os participantes). Cada participante é representado por uma coleção formada por uma identidade, um domínio e um papel. Assim como o gerenciamento de serviços, o contrato pode estar relacionado a uma política macro, estabelecendo as condições comuns, nas quais pode ser realizado. Uma exigência importante se refere ao encapsulamento do contrato, a qual se deve ao fato de um objeto de contrato ser capaz de esconder detalhes da estrutura contratual, como os assinantes e condições relacionadas a eles, apresentando interface demonstrando informações visíveis a um participante e escondidas a outro.

É composto por três fases, denominadas acordo, compromisso e execução:

Acordo: fase na qual os vários atores são envolvidos na formação e definição do conteúdo de contrato, termos e condições.

Compromisso: é a fase correspondente às ações do mundo real, o ato de assinar um contrato (o que não deve ser interpretado como uma exigência para uma assinatura digital).

Execução: fase de execução propriamente dita; corresponde à ação de invocar ou habilitar um contrato.

2.7.6. Módulo IPR – *Intellectual Property Right*

Fornece suporte para a administração e gerenciamento dos direitos de propriedade intelectual, incluindo direitos autorais e de propriedade e possibilita meios de proteger os documentos e módulos presentes na solução de comércio eletrônico, habilitando ou não a transferência e distribuição de conteúdos.

2.7.7. Módulo *Object Browser* – componentes para *Desktops*

O módulo *object Browser* trata da apresentação e administração de componentes como serviços, contratos, certificados, entre outros. É composto basicamente por uma interface gráfica a qual possibilita ao usuário realizar todas as operações referentes às atividades de comércio eletrônico. A OMG propõe que esse módulo deva possibilitar os seguintes procedimentos:

- Possuir um local para armazenamento de endereços de diversos *sites* de comércio eletrônico.
- Realizar a conexão através de diferentes protocolos de comunicação.
- Navegar pelos sites de comércio eletrônico, no estilo *explorer*.
- Visualizar e controlar compras, através de uma ferramenta que demonstre o pedido e realize o pagamento.

2.7.8. Módulo de Agências

O Módulo de Agências fornece suporte às exigências gerais necessárias a padronização da presença de um participante no mercado. Atualmente, não há na Internet nenhum mecanismo para inspecionar um provedor, além do nome da companhia, embutido em um endereço URL (Uniform Resource Locate). O módulo de agências deve estabelecer um ponto de acesso formal e uma interface de pesquisa para um participante do mercado eletrônico.

2.7.9. Módulo de Catálogos

Catálogo é um objeto estruturado capaz de ser inspecionado, navegado e transferido através da rede. O catálogo provê uma interface a qual permite reordenar as suas entradas em diferentes visões, estendendo, modificando ou eliminando seu conteúdo. É fundamental para um catálogo apresentar, em sua estrutura, serviços e informações de contrato. A OMG o distingue em duas formas: objeto e serviço.

O **Catálogo–objeto** é um objeto estruturado que permite ser inspecionado, navegado e transferido através da rede, conforme as descrições acima citadas.

O **Catálogo–serviço** é um serviço (como descrito no módulo de Gerenciamento de Serviços) que retorna um catálogo–objeto.

Exigências básicas do catálogo–objeto:

- Exercer controle sobre a ordenação, adição e remoção de itens.
- Apresentar interface de pesquisa.
- Conter meios para a associação explícita de um catálogo com uma agência.

A infra–estrutura de catálogos deve apoiar a integração transparente entre catálogos de diferentes fontes e possibilitar o surgimento de vários tipos de serviços. Por exemplo, é possível ao usuário, quando consulta dado catálogo de determinado fornecedor, se inscrever (*subscribe*) nele, atrelando-se a esse catálogo e recebendo informações desejadas, sempre ao aparecerem novidades ou sempre ao ser satisfeita alguma condição (como promoções). O catálogo deve manter estrutura a qual determine ao próprio módulo se encarregar de realizar a inscrição, alteração ou cancelamento, conforme as necessidades dos clientes.

3. Tecnologias envolvidas na criação de catálogos eletrônicos

O uso das tecnologias tidas como abertas, voltadas muitas vezes à Internet mas não exclusivamente a essa, permitem o desenvolvimento de aplicações independente de plataformas e, ainda, a integração de sistemas legados. Um claro exemplo é o desenvolvimento de aplicações para serem manipuladas via *browser* padrão, que permitem ao usuário trabalhar com qualquer Sistema Operacional na sua máquina, independemente da plataforma do servidor para qual foi concebida a aplicação.

Com a adoção de padrões abertos, cresce o desenvolvimento e o uso de *software* livre - programas de computador que seguem a Licença Pública Geral (GPL), publicada pela Free Software Foundation, assim considerado porque a licença permite as seguintes liberdades: executar o programa com qualquer propósito; modificar e adaptar o programa às suas necessidades; redistribuir cópias grátis ou com taxas.

O desenvolvimento é utilização de *software* livre manifestam-se como alternativa econômica e tecnológica ao mundo proprietário, o qual dita os parâmetros de custo e desenvolvimento.

3.1. Hipertext Markup Language - HTML

A HTML (*HiperText Markup Language*, ou Linguagem de Formatação de Hipertexto) é uma linguagem de marcação de texto utilizada para a publicação de hipertextos na WEB (*World Wide WEB*), através de páginas que preservam a estrutura do texto nas diferentes plataformas em que são descarregadas [Rocha, 1996]. Não é uma linguagem de programação, apenas uma linguagem de marcação que permite especificar de qual modo serão exibidos os detalhes de um documento, pela inclusão de pequenos códigos (*Tags*). Existe um grupo de trabalho responsável pela padronização da HTML chamado W3C (*World Wide WEB Consortion*).

Na literatura, é encontrado também o termo HTML+, um super conjunto de HTML projetado para permitir um gradual avanço sobre o formato anterior, como a utilização de tabelas, formulários para acesso a banco de dados. O HTML+ foi a minuta que deu origem à versão 3.0 da HTML [Venetianer, 1996].

3.1.1. Pontos importantes a respeito da HTML

Algumas questões referentes a HTML merecem destaque:

- A HTML foi criada para a WEB. Embora a maioria dos processadores de texto tenha muito cuidado em relação a pequenos detalhes, a HTML foi projetada para funcionar com, praticamente, qualquer tipo de computador, em qualquer parte do mundo. Ela tem a finalidade de ser fácil de transportar pela Internet, fácil de ler e compreender por todos os diferentes tipos de computadores.
- A HTML é um padrão aberto. Não é marca registrada de nenhuma empresa, nem pertence a qualquer tipo de organização. É de domínio público.
- A HTML é legível para as pessoas. Ela foi criada desde o princípio de modo a fazer sentido para as pessoas, tornando fácil seu entendimento.
- A HTML incorpora o hipertexto. Ela foi criada de forma a facilitar a elaboração de documentos que podem transportar não somente texto, como também figuras e arquivos multimídia.

Hipertexto é a tecnologia a qual permite marcar palavras ou imagens a fim de servirem de ligação para outros documentos. Isso possibilita a palavras ou imagens marcadas, geralmente sublinhadas e de cor diferente, manterem vínculo com outro documento ou página da WEB, ou seja, ao clicar com o *mouse* sobre elas, um novo documento será demonstrado pelo *browser*. Dessa maneira, o hipertexto torna uma página da WEB mais interativa, permitindo ao usuário "navegar" pelos documentos e, ao mesmo tempo, se transforma na base da organização da WEB, dispensando a forma antiga de organização baseada em menus.

3.1.2. O formato da HTML

Sendo uma linguagem de marcação de texto, a HTML não se preocupa com a aparência do documento, e sim com a sua estrutura [Venetianer, 1996], porém seguir uma certa padronização na confecção do documento auxilia na sua compreensão. A formatação da página é realizada exclusivamente com base na marcação de texto pelos descritores HTML (*Tags*), ou seja, não adianta formatar o texto do documento utilizando–se funções de editores de texto, como tabulação e parágrafos, pois esta

formatação é ignorada pelo *browser* (o *software* que realiza a leitura do documento HTML).

Todo documento HTML apresenta elementos iniciados com o caracter "<" e finalizados com o caracter ">" (descritores iniciais). Esses elementos são as etiquetas ou *Tags* de HTML, ou seja, os comandos de formatação da linguagem. A maioria das *Tags* tem a sua correspondente de fechamento (descritores finais), caracterizados pela barra inclinada antes da *Tag*. <Tag>..... </Tag>

As *Tags* definem a formatação de uma parte do texto, indicando onde começam e onde terminam os textos com a formatação por ela indicada [Rocha, 1996]. Os browsers entendem tudo o que estiver entre essas *Tags* como um elemento de controle de texto, não devendo ser mostrados na tela. Alguns elementos não marcam uma região no texto, apenas inserem algo no documento; estes elementos são considerados "vazios" e são caracterizados pela inexistência da Tag correspondente ao fechamento.

3.1.3. Vantagens e limitações da HTML

A grande vantagem da linguagem HTML, e o principal motivo de sua grande aceitação na WEB, é permitir a qualquer tipo de computador, trabalhando sobre qualquer plataforma, a demonstração de um documento. Além disso, sua forma simples de criação e interpretação, aliada ao surgimento de uma grande variedade de ferramentas para sua manipulação, têm feito a HTML tornar–se um padrão aceito e compreendido mundialmente.

Esse grande sucesso fez com que a HTML fosse a principal forma de distribuição de documentos pela WEB, gerando o surgimento de novas exigências, ou seja, o mercado passou a necessitar de características não suportadas pela HTML. Dentre os vários aspectos, os mais importantes se referem à qualidade da aparência das páginas e à busca e recuperação de documentos.

Para resolver o problema da qualidade da apresentação dos documentos surgiu o padrão CSS (Cascading Style Sheets), conhecidas como folhas de estilo. A CSS permite a criação de estilos uniformes (tamanho, cor e formato das letras, cor da tela de fundo etc.) em páginas WEB. O código pode ser indicado na própria página HTML ou num arquivo externo com extensão CSS. Qualquer alteração na folha de estilo reflete em

todas as páginas do site em que estiver sendo utilizada. Usando uma CSS, os elementos podem ser posicionados na página com precisão, pela utilização das coordenadas a serem ocupadas, possibilitando a criação de camadas distintas de conteúdo, aplicando um texto sobre uma imagem ou sobrepondo dois textos.

Na busca e recuperação de documentos reside o principal problema da HTML. A maioria dos serviços de pesquisa da Internet ignoram a marcação dos documentos e, simplesmente, tratam cada página como um texto livre, sem estrutura. Como resultado disso, quando um serviço de busca faz uma pesquisa, é gerado um número muito grande de itens, a maioria irrelevante ao pedido original. Esse aspecto estimulou o surgimento de um novo padrão: o XML.

3.2. *eXtensible Markup Language* - XML

eXtensible Markup Language - XML foi criada a partir de 1996, por um grupo de 80 peritos em SGML (*Standard Generalized Markup Language*) os quais formaram um consórcio denominado *World Wide WEB Consortium* (W3C), sob a chefia de Jon Bosak, da Sun Microsystems, com a missão de desenvolver uma linguagem de marcação com o poder e a generalidade da SGML e, ao mesmo tempo, pudesse ser usada na WEB. O objetivo de tal trabalho era fornecer, aos usuários da WEB, maneiras de identificar suas próprias *Tags* e atributos quando desejassem, ao invés de usarem o esquema de comandos da HTML.

A XML fornece um meio completo para a autoria e entrega de recursos informativos por toda a WEB, independententemente de plataformas e de sistemas, proporcionando um maior apoio à circulação dos hipertextos, em vista do que a HTML oferece hoje. O grupo encarregado do desenvolvimento da XML tem selecionado somente os recursos da SGML absolutamente necessários para a WEB, o que equivale dizer que a XML é um subconjunto da SGML.

XML é uma evolução da HTML, uma linguagem com a finalidade de descrever informações, possibilitando o desenvolvimento do comércio eletrônico na Internet. Com a utilização da XML, há a possibilidade da distribuição de documentos parciais ou completos na WEB.

Para uma melhor introdução à XML, a comparação de um mesmo texto escrito em HTML e XML:

Em HTML:

<HTML><BODY>

JCN, fevereiro/2001, texto que pretende demonstrar as diferenças entre HTML e XML.

</BODY></HTML>

Em XML:

<!XML>

<TEXTO>

<AUTOR> JCN </AUTOR>

<DATACRIAÇÃO>fevereiro/2001</DATACRIAÇÃO>

<TÍTULO>Diferenças entre HTML e XML</TÍTULO>

</TEXTO>

Analisando pequeno exemplo, nota-se que a XML é parecida com a HTML na utilização de *Tags* para marcar as informações, ressalta, porém, a verdadeira essência da XML: essa linguagem preserva as informações úteis. Ou seja, torna o documento mais "inteligente", criando uma estrutura a qual mostra claramente o significado das informações, pois elas estão agrupadas, e não dispersas pelo documento. Essas novas *Tags* são criadas a critério do desenvolvedor, eliminando a restrição imposta pela HTML, na qual apenas *Tags* definidas pela linguagem podem ser usadas. Sendo assim, as *Tags* são criadas conforme as necessidades dos autores do documento. Um documento XML é composto, basicamente, de três elementos distintos: conteúdo dos dados, estrutura e apresentação.

Conteúdo dos dados: são as informações entre as *Tags*;

Estrutura: a organização dos elementos dentro do documento (pode possuir a estrutura de um memorando, contrato, receita etc.);

Apresentação: é a forma pela qual as informações são apresentadas ao leitor do documento.

A idéia central da XML é adicionar benefícios significativos ao documento concomitantemente à manutenção, separadamente, dos três aspectos mencionados.

Como a HTML, a XML não é um programa, é apenas uma linguagem que permite realizar a descrição, captura, processamento e publicação de informações, oferecendo inúmeras vantagens sobre a HTML, conforme se verá adiante.

As principais características da XML são [W3C, 1997]:

- Ser uma linguagem simples, a qual possibilite rápida construção de documentos, e de fácil implementação na WEB.
- Produzir documentos de acordo com a sintaxe da linguagem SGML.
- Fornecer suporte para *hiperlinks* compatíveis com a abordagem URL.
- Fornecer suporte para uma grande variedade de aplicativos.
- Fornecer um mecanismo de folhas de estilo genérico e poderoso.

A XML é indicada às organizações carentes em disponibilizar recursos de informação diferenciada como: livros, transações financeiras (EDI), manuais técnicos, fórmulas químicas, catálogos de produtos, itens de enciclopédias, entre outros [LIGHT, 1998].

Com adoção da XML como novo padrão, será necessário o desenvolvimento de ferramentas capazes de interpretarem os documentos XML. Os fabricantes de *browser* já estão se adaptando a essa nova tecnologia, possibilitando aos seus produtos, alternadamente, exibirem documentos XML e páginas HTML, como exemplo, o *browser* Internet Explorer a partir da versão 5.0, da Microsoft.

3.2.1. Documentos válidos e bem formados

O documento XML bem formado é aquele cujas *Tags* estão distribuídas corretamente pelo documento, isto é, para toda *Tag* de abertura existe sua correspondente de fechamento (fato nem sempre observado num documento HTML); existe uma *Tag* mais externa (a raiz de todo o documento) na qual todas as outras *Tags* e

dados estão cuidadosamente aninhados. Ou seja, o documento será bem formado quando um processador XML (como um *browser*) tem a possibilidade de criar uma estrutura hierárquica (em forma de árvore). O fato de um documento estar bem formado, não significa ser, o mesmo, válido.

Descreveu-se, acima, sobre a liberdade dos autores de documentos XML em criar suas próprias *Tags*. Supondo, por exemplo, que um autor elabore *Tags* para serem utilizadas em um documento responsável por manter informações sobre livros. O autor cria *Tags* como <AUTOR>, <TÍTULO>, <EDITORA> etc. Dessa forma, é definido um conjunto de *Tags* aparentemente padronizadas, a serem utilizadas toda vez que dado documento for criado. Esse conjunto pode ser armazenado em um arquivo (arquivo DTD, veja item 3.3), mantendo as características de cada uma das *Tags* (conteúdo, obrigatoriedade, seqüência etc.). Quando um processador XML ler tal documento, ele realizará a sua validação (desde que esteja definido no documento e o processador possua essa capacidade), isto é, o procedimento verifica se todas as *Tags* usadas estão de acordo com os padrões previamente definidos pelo autor. Para um documento XML ser válido, obrigatoriamente, ele deve ser bem formado.

3.2.2. As vantagens da XML

Um documento XML traz consigo o conhecimento de sua própria estrutura e semântica, tornando-se muito mais útil do que um documento HTML, iniciando com uma definição de suas próprias regras e requisitos de recursos, armazenadas num arquivo DTD (Document Type Definition).

Graças à XML, que não permite a exclusão de *Tags* de encerramento, a estrutura lógica interna de todos os documentos XML é clara para todos. Portanto, todo documento XML pode, e deve, ser visualizado como uma estrutura hierárquica, cuidadosamente elaborada, contendo dados associados, trazendo benefícios como os descritos a seguir.

As principais vantagens oferecidas pela utilização da XML são:

- Os *browsers*, apoiados por linguagens de programação como Java ou JavaScript, podem fazer grande parte do trabalho de processamento; o conteúdo do documento

pode ser manipulado e reorganizado; cálculos podem ser realizados para gerar novos conteúdos instantaneamente, proporcionando a geração de outros documentos.

- Determinado documento com mesmo conteúdo pode ser visualizado de diferentes formas, para diferentes usuários, pela utilização de folhas de estilo.
- O conteúdo de um documento pode ser pesquisado de maneira inteligente, com base em sua estrutura. Por exemplo, um *browser* pode ser orientado a encontrar quais dos livros, presentes no documento, possuem um determinado autor.
- As ferramentas de criação, baseadas em XML, permitirão aos autores concentrarem–se no conteúdo do documento e não se preocupando com sua formatação.
- Permite um alto grau de automação para agentes de *software*.
- Os documentos XML são, essencialmente, banco de dados de informações, podendo ser consultados e processados como qualquer banco de dados tradicional.
- Possibilita a criação de *Tags* conforme a necessidade.
- A XML permite a transmissão de qualquer tipo de informação estruturada por toda a WEB. Aplicações XML padronizadas possibilitarão o trabalho conjunto de diferentes aplicativos, significando uma maior interoperabilidade.
- Do ponto de vista de um publicador, a informação por ele mantida é o seu "Capital de Giro". A XML permite a adminisração desse bem de maneira coerente e eficiente: grandes recursos de informação podem ser divididos em micro documentos, a serem mantidos e administrados com a utilização de técnicas de banco de dados.
- Os recursos de circulação da XML podem ser usados para criar uma rede WEB de conhecimento (*knowledge* WEB), ou até várias redes superpostas, as quais, de outra forma, constituiriam uma massa de documentos sem conexão. É possível se manter e preservar essas circulações, separadas dos próprios documentos, simplificando a tarefa de manutenção dos links.

- Do ponto de vista do cliente, receber XML em vez de HTML torna–o muito mais auto–suficiente. Quando a informação codificada em XML está relacionada a uma aplicação específica de interesse do cliente, seu valor aumenta dramaticamente. Quando a informação é guardada em uma aplicação XML específica, sua marcação pode refletir precisamente as semânticas daquela informação.

Com todas as características descritas, equivale dizer que pesquisar na WEB tem a tendência de se tornar um processo muito mais preciso do que é atualmente. Sem dúvida, a XML é a nova forma de marcar informação na WEB, não significando o fim da HTML, pois as duas linguagens possuem papéis diferentes, conforme demonstrar-se-á adiante.

3.2.3. Elaboração de um documento XML

A seguir são descritos alguns detalhes de como criar um documento XML. Há sete formas de marcação possíveis de serem incluídas nos documentos XML:

1. *Tags* de início e término.
2. Designações de atributo.
3. Referência a entidades.
4. Comentários.
5. Instruções de processamento.
6. Seções CDATA.
7. Declarações do tipo de documento.

3.2.3.1. *Tags* de início e término

Os elementos são os principais blocos de construção das estruturas hierárquicas da XML. A presença dos elementos em um documento é denotada por *Tags* de diversas formas e a maioria deles deve conter alguma informação: talvez dados de caracteres, talvez outros elementos, ou uma mistura dos dois. Esses elementos têm pontos de início e término denotados por *Tags* de início e término, respectivamente. Exemplos:

Tag	Significado
<Livraria>	Inicia um elemento.
</Introdução>	Encerra um elemento Introdução
<Meu Livro>	Tag de início incorreta. Nomes de tipo de elemento não podem conter espaços.
<42>	Tag de início incorreta. Nomes de tipo de elemento não podem começar com um número.
</ Produto>	Tag de término incorreta. Não são permitidos espaços entre a barra e o nome do tipo de elemento.

Tabela 1 - Exemplos de Tags em XML.

Os elementos podem ser aninhados em uma profundidade arbitrária para descrever estruturas complexas. Aqui está um documento XML com alguns elementos aninhados:

```
<TiposDeAutomóveis>
        <Fabricante>
                <Nome>
                GM
                </Nome>
                <Modelos>
                        <Motor1000>
                                <Descrição>
                                CORSA 1.0 modelo básico, gasolina
                                </Descrição>
                        </Motor1000>
                        <Motor1.6>
                                <Descrição>
                                CORSA GL 1.6, gasolina
                                </Descrição>
                        </Motor1.6>
                </Modelos>
        </Fabricante>
</TiposDeAutomóveis>
```

3.2.3.2.Designações de atributo

Os atributos são informações tipicamente pequenas, associadas aos elementos XML. Os atributos aparecem em diversas formas e tamanhos, e são controlados na DTD, como se verá posteriormente. Entretanto, em um documento XML, todos os

atributos assumem a mesma forma simples. As designações de atributo sempre aparecem na Tag de início de um elemento, neste formato:

[nome do atributo] "=" [valor do atributo]

<Empresa classe="pequena">

Nesse exemplo, o atributo classe está sendo usado para identificar qual o porte de determinada empresa (pequena, média ou grande).

3.2.3.3.Referência a entidades

Entidades são blocos de construção física de documentos XML. Uma entidade é uma unidade de texto – às vezes, tão simples quanto um único caractere, outras tão complexo quanto um documento inteiro. As entidades são incluídas em documentos XML por uma referência à entidade. Quando o interpretador XML encontra uma referência à entidade, ele acrescenta seu conteúdo no documento. As referências à entidade podem ser usadas para acrescentar arquivos inteiros a um documento, conforme abaixo:

```
<Livro>
        &Capitulo1;
        &Capitulo2;
        &Capitulo3;
</Livro>
```

Esse pequeno trecho informa ao processador XML: faça com que três capítulos de um livro, armazenados em entidades separadas, fiquem juntos em um único elemento Livro. As entidades são referenciadas pelo caracter "&".

3.2.3.4.Comentários

Os comentários da XML assumem a seguinte forma:

<! — este é um pequeno comentário — >

Para haver compatibilidade com a SGML, a string "– –" não pode ocorrer com um comentário:

<! – – este não é – – um comentário bem formado em XML – – >

3.2.3.5.Instruções de processamento

Instrução de processamento é um mecanismo utilizado para armazenar informações específicas de um documento XML. A idéia é fornecer uma maneira de permitir a um documento XML a possibilidade de comunicação com um determinado aplicativo. A XML faz uso de uma instrução de processamento conhecida como declaração da XML. Essa é uma instrução de processamento a qual deve iniciar todos os documentos XML, conforme abaixo:

<?xml version = “1.0”?>

Ao encontrar tal instrução delimitada pelos caracteres “?”, um aplicativo será informado que o documento foi elaborado segundo os padrões presentes na versão 1.0 da XML.

3.2.3.6.Seções CDATA

Às vezes, um documento deve conter grandes números de caracteres considerados especiais por um interpretador XML, caracteres como “<” e “&” entre outros. A XML permite o isolamento de um bloco de texto da atenção do interpretador, usando uma seção CDATA, conforme abaixo:

<Documento>

<![CDATA[if a<b and b<c then a<c]]>

</Documento>

Em circunstâncias normais, o sinal “<” no documento acima, atrairia a atenção do interpretador (ele entenderia que o sinal corresponde ao início de uma Tag). Colocando a string “<!CDATA[“ como prefixo e complementando com “]]>”, a seção inteira é isolada, tornando, os caracteres reservados, caracteres simples.

3.2.3.7.Declarações do tipo de documento

Para se associar um documento XML a uma DTD, utiliza-se uma declaração do tipo de documento no documento XML, como o exemplo abaixo:

<!DOCTYPE comercio SYSTEM “comercio.dtd”>

Detalhes minuciosos sobre DTD serão vistos a seguir no item 3.3.

3.2.3.8.Um documento XML completo

Já foram analisados todos os detalhes necessários para a criação de um documento XML. Apresentar-se-á, a seguir, um pequeno exemplo de um documento XML para a área de computadores.

```
<?xml version = “1.0”?>
<!DOCTYPE agenda SYSTEM “agenda.dtd”>
  <Agenda>
        <!—Minha agenda de trabalho—>
        <Ontem>&Ontem</Ontem>
        <Amanha>&Amanha</Amanha>
        <Compromisso Prioridade="Normal">
              <Hora>9:00</Hora>
              <Descrição>Encontrar com Carlos e falar sobre
    música</Descrição>
              <Local>Escritório do Carlos</Local>
        </Compromisso>
        <Compromisso Prioridade="Baixa">
              <Hora>12:30</Hora>
              <Descrição>Almoçar com Ivone</Descrição>
              <Local>Restaurante Casa Blanca</Local>
        </Compromisso>
        <Compromisso Prioridade="Urgente">
              <Hora>14:45</Hora>
              <Descrição>Encontrar com agente de vendas</Descrição>
              <Local>Rua 24 de maio, 1835</Local>
        </Compromisso>
```

```
<Compromisso Prioridade="Normal">
        <Hora>16:40</Hora>
        <Descrição>Conferência com Bill Gates</Descrição>
        <Local>Câmara Municipal</Local>
  </Compromisso>
<Compromisso Prioridade="Alta">
        <Hora>18:25</Hora>
        <Descrição>Levar crianças ao cinema</Descrição>
        <Local>Shopping Iguatemi</Local>
</Compromisso>
</Agenda>
```

3.2.3.9.Novas formas de pesquisa com XML

Páginas WEB, escritas em HTML, incluem informação referente à formatação do texto. Mas a HTML não permite um modo para descrever os conteúdos do texto: o significado está perdido porque não há nenhum modo para demarcar isto. Torna–se necessária a criação de páginas dinâmicas as quais possibilitem escolher a forma de apresentação de acordo com a necessidade do usuário, por exemplo, listas ordenadas pelo nome, cor, preço, fabricante, e assim por diante. A XML torna isso uma realidade.

Com o uso da XML é possível tornar o serviço de busca mais "inteligente": em vez de procurar por uma palavra em todo o texto de uma página, pode ser usada uma *Tag* para especificar em quais partes da página deve se efetuada a pesquisa, evitando–se páginas irrelevantes, sendo fornecida uma listagem com informação mais coerente.

Por exemplo, ao procurar Albert Einstein num serviço de busca tradicional, serão encontradas faculdades, conferências universitárias que o mencionam, vínculos para sites sobre ele, história Judia, páginas de cotações, biografias dos colegas dele, e assim por diante. Talvez o usuário esteja interessado apenas em conhecer quais livros e artigos ele escreveu. Seria melhor se o serviço de busca possibilitasse a pesquisa em todas as

páginas onde Albert Einstein fosse o autor, com isso, todas as páginas irrelevantes não apareceriam, facilitando o trabalho de quem solicitou a pesquisa.

A WEB está se tornando um tipo de inteligência *cyborg*: humano e máquina juntos para gerar e manipular informação. Ela está possibilitando ao homem eliminar o trabalho braçal que envolve a troca e manipulação de conhecimento [Dertouzous, 1997].

A troca de marcadores estruturais (como é o caso de HTML) para marcadores semânticos (como é o caso de XML) é uma fase crítica na luta para transformar o grande número de informações numa cadeia de conhecimento universal [Adams, 1979].

As buscas, como já citado, podem ser realizadas de várias formas. Supondo o caso de uma biblioteca, as buscas poderiam ser realizadas pelo nome do autor, títulos, tema, assuntos dentro de um tema. Essa customização provém de funcionalidades que usufruem das características fornecidas pela XML, proporcionando a uma ferramenta de busca fornecer o resultado de uma pesquisa com as seguintes vantagens:

- Reindexação dos elementos encontrados de maneira mais intuitiva.
- Apresentação da informação em diferentes formatos, como imagem/vídeo ou texto.
- Fornecimento da informação em diferentes graus de profundidade de conteúdo, possibilitando ao documento ser apresentado em diferentes idiomas.
- Adaptação do layout. Os documentos podem ser recebidos de maneiras diferentes de acordo com os recursos presentes na máquina do usuário .
- Processo de busca manipulável de acordo com parâmetros do usuário. O usuário é responsável por definir o formato de apresentação do documento a ser recebido como resultado da pesquisa.

Apesar de já ser possível tal procedimento, ele está longe se tornar realidade. Todos os documentos da WEB e todas as informação neles contidas deveriam ser escritas em um formato padronizado, ou seja, em XML. São necessários investimentos para se padronizar os diversos ramos de atividade, pois existe grande quantidade de dados circulando na WEB. Apesar disso, nada impede que grupos fechados se unam para aproveitar os recursos dessa nova tecnologia.

3.2.4. Pontos chave

Os itens a seguir descrevem a estrutura necessária para que a XML se torne uma realidade e possa resolver os problemas de busca [SearchTools, 2000].

- A maioria dos *browsers* precisam reconhecer páginas em XML. Enquanto isso não acontecer, a maioria dos sites precisará possuir versões em HTML e XML de suas páginas. O fato de a maioria dos *browsers* não exibirem XML não deveria restringir a existência de um sistema para procurar páginas em XML, desde que fosse possível aos resultados serem gerados em XML ou HTML, de acordo com a versão do *browser* ou preferência do usuário.

- Cada segmento da indústria deverá montar sua estrutura padrão e haverá batalhas a respeito de quem irá controlar esse padrão. Todos os ramos de atividade deverão padronizar seus documentos, determinando a estrutura adequada para eles. Para alguns grupos isso é simples, por exemplo, sumário de livros, ou conteúdo de disciplinas. Outros grupos como matemáticos e químicos já estão desenvolvendo padrões para seus documentos.

- Provedores de conteúdo de WEB Sites terão de colocar as *Tags* nas páginas, de acordo com as estruturas padrão de cada um. Colocar as *Tags* no texto é um procedimento difícil, porque exige dos escritores e editores do documento total compreensão do contexto dos dados e onde tais dados se localizam em sua estrutura. Serão necessárias boas ferramentas XML, como editores e gerenciadores de banco de dados. Quando há opção de exportar o conteúdo do documento do bancos de dados e de etiquetar tal documento automaticamente, o processo é muito mais simples. Porém, há milhões de páginas de HTML, muitas das quais incluem dados úteis que nunca serão tocados novamente.

- Serviços de busca deveriam manter as informações pesquisadas com metadados (veja item 3.5), bem como o índice deve manter as informações de busca como metadados. Além de *Tags* básicas, os índices deveriam armazenar a hierarquia inteira, de forma a possibilitar ao pesquisador especificar a tag certa até mesmo quando os nomes de *Tags* se repetem em uma estrutura. Por exemplo, um documento de listas de trabalho que apresente uma tag <locate> para a sede de

companhia e uma diferente <locate> para o local de trabalho, e ainda outra tag <locate> para o escritório do chefe. Obviamente, estas *Tags* são muito diferentes e deveriam ser procuradas separadamente.

- Interfaces de procura terão de exibir as opções de um modo útil, disponibilizando a informação estrutural aos pesquisadores. Um menu tipo popup com todas as possíveis formas de pesquisa talvez seja útil. Procurar apenas campos em lugar do documento inteiro é mais difícil e requer mais esforço do usuário, o qual tem de se adaptar à estrutura dos documentos. A interface está mais como uma máquina de procura de banco de dados do que um campo de procura simples.

Nota–se, portanto, que a XML não resolverá de uma hora para outra os problemas de busca pela Internet, porque muitos padrões (como citados nos itens 2.6.1 e 2.6.2.) devem emergir e os diversos segmentos do mercado necessitam fazer grandes investimentos. Porém, com o decorrer do tempo, isso poderá trazer grandes benefícios.

3.2.5. Busca Convencional X Busca Otimizada

A tabela 2 procura demonstrar as principais diferenças existentes entre os serviços de busca convencionais, realizados constantemente através de páginas WEB escritas em HTML, em relação à pesquisas otimizadas, possíveis de serem realizadas através da utilização da linguagem XML.

Busca convencional	Busca Otimizada
A ferramenta de busca armazena a palavra a ser pesquisada como um índice e procura nos diversos documentos a sua ocorrência.	A ferramenta de busca deve reconhecer toda a estrutura do documento, identificando cada Tag como um objeto a ser manipulado.
Os dados do documento são todos do mesmo tipo.	Um documento pode conter diferentes tipos de dados, como um único campo ou como um registro composto por campos, podendo retornar vários registros.
Retorna uma lista de documentos com alguma informação sobre eles como resultado da pesquisa.	Retorna uma lista de registros, podendo pertencer a vários documentos diferentes e gerando o documento resultante da pesquisa, normalmente no formato de uma tabela.
Realizam apenas a pesquisa em um índice previamente definido.	Além de realizarem a pesquisa através de múltiplas fontes, podem realizar a atualização de documentos.

Tabela 2 - Diferenças entre a busca convencional e a otimizada.

Com base em todos os aspectos estudados sobre XML, nota-se que tal linguagem possui a habilidade de descrever o significado dos dados em um documento ou mensagem, propiciando o desenvolvimento de padrões, nos diversos segmentos do mercado, tornando a comunicação mais eficiente e compreensível entre parceiros de negócio e contribuindo para o desenvolvimento do comércio eletrônico.

3.3. *Document Type Definition* - DTD

3.3.1. Definição DTD

DTD (*Document Type Definition*) é um arquivo que trabalha em conjunto com documentos XML, cuja função é armazenar todas as *Tags* porventura contidas em um determinado documento XML. Com o DTD é possível validar um documento XML, ou seja, verificar se ele foi criado, corretamente, segundo um padrão definido. Um DTD contém informações necessárias para se criar um documento XML válido, assim como para realizar seu processamento. Sem essa informação, leitores do documento XML (como *browser*) poderiam não saber como processar links, imagens ou entidades, e os autores dos documentos não teriam um padrão a seguir no momento da elaboração. Antes de elaborar um DTD, deve–se possuir um entendimento muito claro a respeito do tipo de documento criado.

Com a utilização do DTD se pretende estabelecer padrões de marcação para as mais diversas áreas, possibilitando a todas as empresas de um mesmo ramo a divulgação de seus produtos e serviços de uma maneira unificada. Como exemplo, algumas instituições já estão padronizando seus produtos, criando, a partir da linguagem XML, novas sub–linguagens especializadas em alguma área, como é o caso de MathML (marcação padronizada para fórmulas e equações matemáticas) [Light, 1998].

3.3.2. Utilização da DTD em um documento XML

Para se associar um documento XML a um DTD, é usada uma declaração do tipo de documento no documento XML, como o exemplo abaixo:

<!DOCTYPE comercio SYSTEM “comercio.dtd”>

Isso quer dizer: o DTD do presente documento está disponível no documento comercio.dtd. É um exemplo que possibilita a associação de um documento XML a um

DTD externo. Há a possibilidade da descrição do DTD estar interna no próprio documento XML.

Para poder validar um documento XML, um interpretador de validação da XML deve conhecer três princípios de cada elemento:

- Como o tipo de elemento é nomeado.
- Que tipo de informação pode conter cada elemento.
- Quais atributos estão associados a cada elemento.

3.3.3. Detalhes para a construção de DTDs

A seguir, são apresentados os detalhes necessários para a compreensão e elaboração de DTDs, demonstrando as declarações de elementos, atributos, entidades e suas respectivas interpretações.

Declaração	Interpretação
<!ELEMENT contato (nome, endereço, telefone)>	Um elemento do tipo contato deve conter 3 sub-elementos: nome, endereço e telefone, exatamente nessa ordem.
<!ELEMENT contato (nome, endereço?, telefone)>	Um elemento do tipo contato pode conter 3 sub-elementos: o elemento nome, seguido normalmente (?) de um elemento endereço, por último, contém o elemento telefone.
<!ELEMENT fruta (maça|laranja)>	Um elemento do tipo fruta contém um único Elemento maça ou (|) um único elemento laranja.
<!ELEMENT fruta (maça|laranja)+>	Um elemento do tipo fruta contém um ou mais (+) sub-elementos que são elementos maça ou elementos laranja.
<!ELEMENT fruta (maça|laranja)*>	Um elemento do tipo fruta contém zero ou mais (*) sub-elementos que são elementos maçã ou elementos laranja
<!element Estoque EMPTY>	Isso é um erro. A palavra–chave "ELEMENT" sempre deve estar em letras maiúsculas. Isso se aplica a todas as palavras–chave da XML>
<!ATTLIST produto nome=... >	Um elemento do tipo produto tem um atributo conhecido como nome.
<!ATTLIST produto nome CDATA...>	Um elemento do tipo produto tem um atributo conhecido como nome, cujo valor pode ser uma string de caracteres (exceto

	por três que têm um significado especial para o interpretador: "<", ">" e "&").
<!ATTLIST produto nome CDATA..cor (vermelho/verde)...>	Um elemento do tipo produto tem dois atributos conhecidos como nome e cor. O valor de atributo nome pode ter uma string de caracteres, salvos os casos especiais: "<", ">" e "&". O valor do atributo cor deve ser a string "vermelha" ou a string "verde".
<!ATTLIST produto código ID...>	Um elemento declarado do tipo produto tem um Atributo conhecido como código. Os valores do atributo código devem ser exclusivos entre atributos do tipo ID, em todo o documento XML.
<!ATTLIST produto nome #REQUIRED>	Um elemento do tipo produto tem um atributo conhecido como nome, cujo valor deve ser fornecido.
<!ATTLIST produto nome CDATA "Nobre">	Um elemento do tipo produto tem um atributo conhecido como nome. O valor do atributo nome pode conter uma string de caracteres; na ausência de um valor, use o valor default "nobre".
<!ATTLIST produto cor (vermelha/verde) #IMPLIED >	Um elemento do tipo produto tem um atributo chamado cor. O atributo cor deve ter a string "vermelha" ou "verde". Se um valor não for fornecido, deixe que o aplicativo XML decida o que fazer.
<!ATTLIST produto nome CDATA #FIXED "nobre">	Um elemento do tipo produto tem um atributo conhecido como nome cujo valor deve ser "nobre". Outro valor é um erro.
<!ENTITY dis "Minha dissertação">	Existe uma entidade conhecida como dis. Quando mencionada em um documento XML, o interpretador irá inserir o texto de substituição "Minha dissertação".
<!ENTITY cap1 SYSTEM "http://Web.xyz.com/cap1.xml">	Existe uma entidade conhecida como cap1 Quando mencionada em um documento XML, o interpretador irá inserir o conteúdo do arquivo armazenado no endereço http://Web.xyz.com/cap1.xml

Tabela 3 - Algumas convenções utilizadas na DTD

O DTD abaixo descreve a estrutura de um livro de romance. O elemento principal desse documento é romance (elemento root), definido na primeira linha. Todo DTD, assim como um documento XML, possui um elemento principal o qual engloba todos os outros. O DTD define todos os elementos que possui, seu nome, conteúdo, obrigatoriedade e atributos, conforme especificados na tabela 3.

<!ELEMENT romance (resumo,capítulo+,biografia?,crítica*)>

<!ELEMENT prefacio (parágrafo+)>

<!ELEMENT capítulo (título,parágrafo+,seção+)>

<!ELEMENT seção (título,parágrafo+)>

<!ELEMENT biografia (título,parágrafo+)>

<!ELEMENT crítica (título,seção+)>

<!ELEMENT parágrafo (#PCDATA|palavrachave)*>

<!ELEMENT título (#PCDATA|palavrachave)*>

<!ELEMENT palavrachave (#PCDATA)>

3.3.4. Validação de um documento XML com o DTD

Quando um processador XML realiza a leitura de um documento, ocorre o processo de validação (desde que exista uma declaração ao DTD), no qual todos os elementos contidos serão analisados, verificando se estão descritos no DTD e quais suas condições de existência. O processo de validação é realizado por um *parser*. A fig. 7 mostra como um *Browser* (por exemplo, o Internet Explorer 5.0) realiza a validação de um documento XML

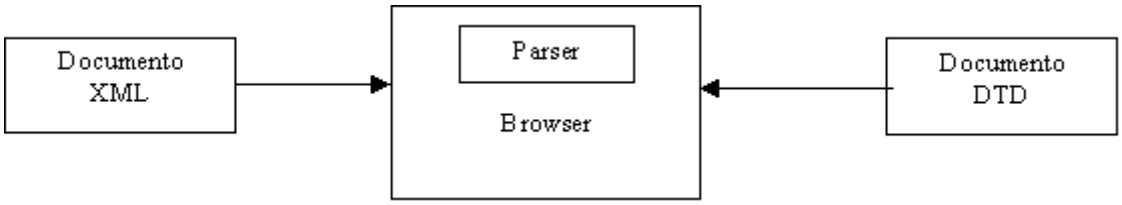


Figura 7 - Validação de um documento XML

3.4. *eXtensible Style Language* - XSL

Em agosto de 1997, a Microsoft, a Inso, a Arbortex e outras empresas colocaram uma especificação preliminar de uma linguagem de estilo para XML conhecida como XSL – *eXtensible Style Language* (linguagem de estilo extensível). O interesse pela XSL tem sido muito grande, com pelo menos três implementações parciais. A finalidade da XSL é fornecer uma folha de estilo poderosa e com uma sintaxe de fácil uso para expressar como os documentos XML devem ser apresentados [Light, 1998].

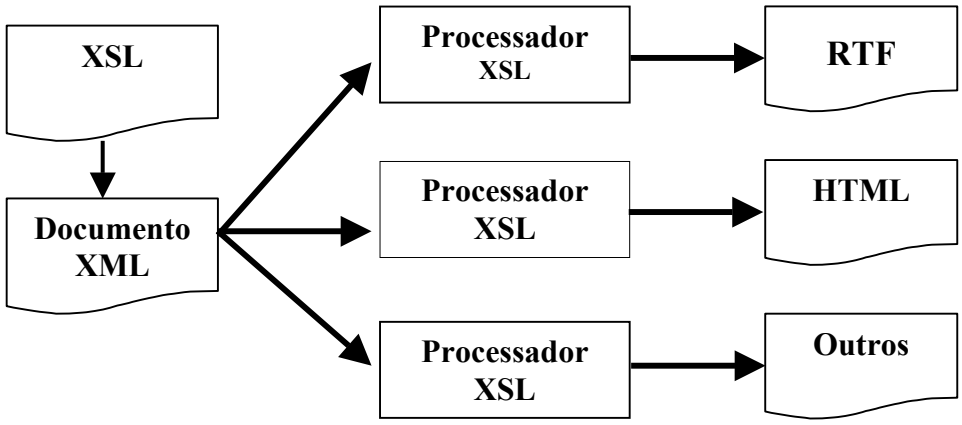


Figura 8 - Geração de várias apresentações a partir de um mesmo documento XML

Uma única folha de estilo XSL pode ser usada para converter um documento XML em diferentes formatos, como RTF (Rich Text Format), HTML, ou qualquer outra forma de padronização existente, usando ferramentas compatíveis com XSL, conforme ilustrado na fig.8, acima.

Para especificar que uma folha de estilo XSL deve ser usada para apresentar um documento XML, é necessário vincular os dois documentos, adicionando, no início do documento XML, uma referência à folha de estilo XSL, como acontece com o DTD, conforme exemplo a seguir:

<?xml–stylesheet href="construlár.xsl" type="text/xsl"?>.

Considerando um dado formato, HTML por exemplo, podem ser desenvolvidas várias folhas de estilo, de forma que o mesmo documento XML possa ser apresentado em várias maneiras diferentes.

A XSL possui recursos para processar um documento XML, ou seja, reorganizar (apagar, incluir, alterar) a geração de conteúdo "em paralelo" ao processo de formatação, característica não suportada por sua antecessora, a linguagem CSS, usada para formatar os documentos HTML. Para aumentar o poder da XSL, são adicionados recursos de uma linguagem de programação, ou seja, divisões por módulos, execuções de loop, variáveis e assim por diante. A XSL incorpora a versão padronizada internacionalmente do JavaScript conhecida como ECMAScript (ECMA–262),

possibilitando a realização de cálculos complexos, ou qualquer manipulação dos objetos da página, de acordo com o necessário.

3.4.1. Princípios do modelo XSL

Da mesma forma que a XML, os desenvolvedores da XSL trabalharam com um conjunto de princípios essenciais, cujos principais são: [W3C, 1997]:

- A XSL deve ser utilizada na Internet .
- A XSL deve ser expressa com a sintaxe da XML.
- A XSL deve fornecer uma linguagem declarativa, realizando todas as tarefas comuns de formatação.
- A XSL deve possibilitar o uso de uma linguagem de script, a fim de acomodar tarefas de formatação mais sofisticadas, além de permitir extensibilidade.
- As folhas de estilo XSL devem ser de fácil criação e legibilidade.

3.4.2. A arquitetura XSL

Uma folha de estilos XSL é composta de um conjunto de regras conhecidas como regras de construção, as quais especificam como um documento XML é convertido em uma hierarquia de objetos, conhecida como objetos de fluxo. Os objetos de fluxo comuns incluem uma página, um parágrafo, uma célula de tabela. Os objetos de fluxo podem se aninhar, ou seja, um objeto de fluxo de página pode conter um objeto de fluxo de tabela que contenha objetos de fluxo de célula. A XSL define um conjunto principal desses objetos de fluxo, comum a todas as implementações XSL. Os objetos de fluxo possuem propriedades associadas conhecidas na XSL como características, responsáveis pelo controle sobre as operações exatas do objeto de fluxo. Exemplos de características incluem largura da página (referente a um objeto de fluxo de página), alinhamento (para objeto de fluxo de parágrafo) etc. Da mesma forma que os objetos de fluxo principais são independentes de qualquer dispositivo de saída, a XSL inclui objetos de fluxo especificamente os quais visam obter valores resultantes em HTML, como BODY, HEAD, TABLE. Como as regras de construção, a XSL oferece regras de estilo responsáveis por definirem as características de formatação, podendo ser aplicadas

várias regras de estilo ao mesmo documento. Uma folha de estilo XSL equivale a um documento XML baseado em um pequeno DTD, com alguns tipos de elementos como <xsl>, <rule> etc. Os objetos de fluxo HTML em XSL também são representados através de *Tags* como <HTML>, <P>, <TABLE>. Essas *Tags* de objeto de fluxo são sempre em maiúsculas (caixa alta), ao passo que as *Tags* da folha de estilo XLS estão sempre em minúsculas (caixa baixa).

A seguir, será dado o exemplo de uma folha de estilos XSL (reconhecida pelo Internet Explorer 5.0), geradora de um documento HTML a partir de um documento XML, com os comentários mais significativos:

```
<?xml version='1.0'?>

<xsl:stylesheet xmlns:xsl="http://Web.w3.org/TR/WD–xsl">
```

Essa linha indica que este documento é um arquivo XSL e o endereço indica a localização do namespace (verifica a validade das propriedades) reconhecido pelo Internet Explorer 5.0

```
<xsl:template match="/">
```

Indica que esse modelo realizará a leitura de todo o documento XML a partir do elemento principal (root).

```
    <HTML>

    <BODY>

    <TABLE BORDER="2">

        <TR>

        <TD>Código</TD>

        <TD>Nome</TD>

        <TD>Preço</TD>

    </TR>
```

Comandos HTML (criam o cabeçalho de uma tabela).

```
    <xsl:for–each select="produto/estoque">
```

```
<TR>
        <TD><xsl:value–of select="codigo"/></TD>
        <TD><xsl:value–of select="nome"/></TD>
        <TD><xsl:value–of select="preço"/></TD>
</TR>
```

Realiza a leitura dos sub-elementos contidos entre as *Tags* abaixo:

```
<produto>
    <estoque>
    </estoque>
</produto>
```

Para cada elemento (código, nome, preço) encontrado entre as *Tags* acima, é criada uma célula da tabela HTML. Sendo assim, esse arquivo XSL cria uma tabela HTML a partir de um documento XML.

```
</xsl:for–each>
</TABLE>
</BODY
</HTML>
</xsl:template>
</xsl:stylesheet>
```

3.5. *Resource Description Framework* - RDF

Houve um aumento exponencial de páginas na Internet e, como conseqüência, o volume de informações tornou-se extraordinariamente grande, aspecto o qual tem dificultado a busca de informação. Dessa forma, torna–se cada vez mais difícil encontrar a informação desejada utilizando–se os atuais serviços de pesquisa.

A saída para esse problema é padronizar a forma de criação de páginas WEB, descrevendo informações sobre seu conteúdo, tornando possível aos sistemas entenderem o conteúdo da página (chamados de “machine–understandable”). Nesse contexto, o papel das páginas WEB é fornecer informações que possibilitem a um sistema identificar e entender o relacionamento existente entre páginas WEB, criando links entre elas de forma automática.

Para tornar possível o entendimento do conteúdo de páginas, com uma informação mais descritiva, utilizam–se metadados. Membros da W3C iniciaram, há mais de dois anos, a padronização do significado para troca de descrições entre *machine-understandables* na WEB, ajudando usuários a encontrarem informação de forma mais eficiente do que o proporcionado pelos serviços de busca atuais, resultando na criação de uma nova tecnologia de busca chamada RDF – *Resource Description Framework*. RDF fornece uma forma simples de fazer declarações sobre resources disponíveis na WEB.

RDF é um modelo de estrutura para metadados; possibilita interação entre aplicações que trocam informação na Rede. O propósito mais amplo de RDF é descrever formas para facilitar o processamento automático de *resources* na rede utilizando metadados e criando maior precisão na localização de documentos com conteúdos específicos, pelo reconhecimento do conteúdo.

3.5.1. O modelo de dados RDF

O modelo básico de dados consiste em três tipos de objetos:

Resources (Recursos): em RDF, um *resource* pode ser uma página WEB, como o documento HTML " http://w3.org/Overview.html ", pode ser parte de uma página, como um elemento específico dentro de um documento HTML ou XML. Um *resource* pode ser também um site da WEB, ou até um objeto, não diretamente acessível pela rede, como um livro impresso.

Properties (Propriedades): uma propriedade é um aspecto específico, característica, atributo, ou relação, usada para descrever um *resource*. Cada propriedade tem um significado específico o qual define seus valores permitidos, os tipos de *resources* possíveis de serem descritos, e sua relação com outras propriedades. Ao conjunto das definições dessas propriedades é dado o nome de Schema. Um schema é o lugar onde são documentadas definições e restrições do uso para propriedades.

Statements (Declarações): um *resource* específico, junto com uma propriedade nomeada, mais o valor dessa propriedade para o *resource* é um *statement* de RDF. São chamadas estas três partes individuais de uma declaração, respectivamente, assunto, predicado e objeto (*subject, predicate, and object*). O objeto de uma declaração (o valor

da propriedade) pode ser outro *resource*, um endereço da WEB, uma simples string ou outro tipo de dado definido pela XML.

Exemplo da declaração de um *resource*:

Nobre é o criador do *resource* http://Web.xyz.com.br.

Subject (Resource)	http://Web.xyz.com.br
Predicate (Property)	Criador Nobre
Object	(literal)

Quadro 1- Declaração de um resource

Nesse caso, o *resource* http://Web.xyz.com.br possui uma propriedade Criador, cujo valor é Nobre.

3.5.2. Exemplo da utilização de RDF

```
<? xml version="1.0" ?>
<RDF xmlns = "http://w3.org/TR/1999/PR–rdf–syntax–19990105#"
xmlns:DC = "http://purl.org/DC#" >
<Descrição about = "http://dstc.com.au/report.html" >
<DC:Titulo> O futuro dos Metadados </DC:Titulo>
<DC:Criador> Mônica Libman </DC:Criador>
<DC:Data> 01–01–1998 </DC:Data>
<DC:Assunto> Metadados, RDF, Dublin Core </DC:Assunto>
</Descrição>
</RDF>
```

Quadro 2 - Exemplo de RDF

A primeira linha desse exemplo indica ser ele o documento XML. A linha seguinte indica dois *namespaces*: o padrão RDF e DC. Todas as propriedades de Descrição terão um desses *namespaces*. A principal seção desse exemplo, entre as *Tags* <Descrição>, mostra quatro propriedades as quais descrevem o *resource*, apontado pelo atributo *about*, definidas pelo *namespace* Dublin Core [DC, 1998]. Nesse caso, as propriedades Titulo, Criador, Data e Assunto são indicadas para o *resource* "http://dstc.com.au/report.html".

3.5.3. Vantagens do RDF

Uma grande vantagem da utilização de RDF é a possibilidade de relacionar valores que representem não apenas uma palavra como também um outro *resource*. Se o conteúdo da Tag Autor fosse um endereço de uma página da WEB, outras informações poderiam ser obtidas a respeito do autor (nome, endereço, telefone etc.), ou seja, informações contidas em outra página da WEB poderiam ser lidas e demonstradas.

Por essa simples descrição de RDF, verifica-se a possibilidade do surgimento de uma nova forma de pesquisa pela WEB, tornando as buscas mais "inteligentes" e eficazes. Com RDF, ao entrar em um página WEB, automaticamente podem ser demonstrados links de assuntos relacionados, de interesse do usuário. Autores e publicadores da WEB possuem uma nova ferramenta para descrever com precisão os materiais que estão disponibilizando na rede. Serviços de busca usarão essa informação para fornecer respostas mais precisas, possibilitando o surgimento de "agentes inteligentes" (*smart agents*).

3.6. *Document Object Model* - DOM

Document Object Model (DOM) é interface neutra a qual permite às linguagens de programação acessarem dinamicamente e atualizarem o conteúdo, a estrutura e estilo de documentos [W3C, 1997]. DOM é uma API (*application programming interface*) para documentos HTML e XML definidora de uma estrutura lógica de um documento e do modo como ele é acessado, manipulado e demonstrado.

Um dos objetivos do DOM é ser uma interface de programação passível de ser usada em uma grande variedade de ambientes e aplicações, isto é, as aplicações podem ser movidas para qualquer plataforma compatível com DOM, sem ser necessária a alteração do código. Conforme citado anteriormente, um documento XML pode ser utilizado para representar muitos tipos diferentes de informação e ser armazenados em diversos sistemas. DOM é uma opção para administrar estes dados. Com o DOM é possível construir documentos, navegar por sua estrutura, acrescentar, modificar, ou apagar elementos e seus conteúdos.

Todo documento da WEB é composto por uma variedade de *Tags* como <B>, <IMG>, <FORM>. O DOM interpreta cada Tag como um objeto, descrevendo cada

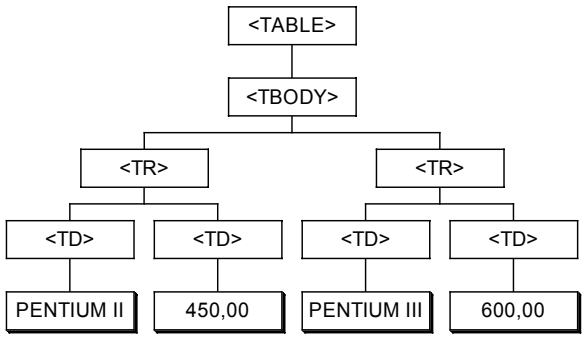
documento como uma coleção de objetos individuais. Cada objeto particular pode ter propriedades associadas a ele, tipicamente, na forma de atributos. Por exemplo, o elemento parágrafo tem um atributo de alinhamento, o qual pode ser fixado para esquerdo, direito ou centro. No modelo de objetos, este atributo é chamado de propriedade do objeto. Um objeto pode ter métodos a ele associados e eventos capazes de afetá–lo. Uma *Tag* de imagem pode disparar um evento, ativado quando o usuário coloca o cursor sobre a imagem. É possível a um desenvolvedor usar o modelo de objeto para encontrar uma imagem em uma página e substituí–la por outra, quando um usuário passar o cursor sobre ela. Enfim, com o DOM consegue–se controle sobre todos os elementos do documento.

3.6.1. O Modelo DOM

Ao interpretar um documento XML ou HTML (bem formado), DOM gera uma representação em memória, criando um modelo de objetos que segue a estrutura do documento, conforme demonstrado a seguir.

```
<TABLE>
        <TBODY>
        <TR>
                <TD>Pentium II</TD>
                <TD>450,00</TD>
        </TR>
        <TR>
                <TD>Pentium III</TD>
                <TD>600,00</TD>
        </TR>
        </TBODY>
</TABLE>
```

Representação de uma árvore no Dom



DOM cria uma representação de objetos em forma de uma árvore. Uma propriedade importante do modelo é o isomorfismo estrutural: se duas interfaces diferentes DOM são usadas para criar uma representação do mesmo documento, elas criarão o mesmo modelo de estrutura, com justamente os mesmos objetos e relações [W3C, 1997].

3.6.2. Vantagens do DOM

HTML possibilita a inserção de cabeçalhos, parágrafos, links hipertexto e outros componentes. DOM é um padrão de representação da estrutura de um documento e torna possível fazer o acesso a esses componentes e manipular seu conteúdo, atributos e estilo. Algumas da vantagens do DOM incluem:

- Interface independente da linguagem e implementação: uma importante característica do DOM é ser uma interface independente, podendo ser usada numa grande variedade de ambientes e aplicações.
- Suporte para HTML e XML: o DOM é designado para documentos HTML e XML. Qualquer elemento encontrado nesses documentos podem ser acessados, mudados, excluídos ou adicionados.
- Suporte as linguagens Java e ECMAScript: o DOM fornece uma interface consistente para a manipulação de um documento através de linguagens de programação como Java e ECMAScript. A linguagem ECMAScript é constituída por vários objetos específicos e a linguagem Java é constituída por uma série de classes que fornecem a funcionalidade exposta pelas interfaces.
- *Style Sheets Support*: o DOM especifica uma forma de manipular e mudar o estilo do documento utilizando as folhas de estilo CSS e XSL.

3.7. *Simple* API for XML - SAX

Simple API for XML (SAX) é um modo alternativo de trabalhar com os conteúdos de documentos XML. É um padrão de fato, foi desenvolvido por David Megginson e outros membros da lista de correio do "XML-Dev".

SAX foi definida para a linguagem JAVA.

Existem dois tipos principais de APIs para XML, de acordo com [MEGGINSON, 1998] :

- API baseada em objetos - DOM;
- API baseada em eventos, SAX.

Uma API baseada em evento informa através do *parser* a ocorrência de eventos (como por exemplo, início e fim de elementos) diretamente para a aplicação por *callbacks*, e normalmente não constrói uma árvore interna. A aplicação implementa *handler*s para lidar com os eventos diferentes, muitos dos quais são tratados pela interface de usuário.

3.7.1. O funcionamento da SAX

Um *parser* SAX lê o documento XML e notifica a aplicação a existência de eventos de seu interesse. Por exemplo, o *parser* pode ler uma *Tag* do elemento do documento XML, e então notifica a aplicação a existência do evento startElement. Após a conclusão do evento o *parser* não necessita manter a *Tag* e a informação associada em memória.

Um *parser* SAX usa alguns callbacks pré-definidos para notificar a uma aplicação sobre a análise de eventos. Este callbacks estão definidos como métodos de interface SAX padrão para ser implementado pela aplicação.

O modelo SAX gera represantação da informação de acordo com a presença de eventos de interesse da aplicação. Com exemplo cita-se a necessidade de encontrar num documento XML na *Tag* <TD> o conteúdo Pentium II.

Abaixo tem-se a representação baseada em evento - SAX:

<TABLE>	Start table
<TBODY>	Start element: tbody
<TR>	Start element: tr
<TD>Pentium II</TD>	Start element: td
<TD>450,00</TD>	Characters: Pentium II
</TR>	End element: td
<TR>	Start element: td
<TD>Pentium III</TD>	Characters: 450,00
<TD>600,00</TD>	End element: td
</TR>	…
</TBODY>	
</TABLE>	End table

3.7.2. Eventos SAX

Os eventos SAX são definidos como métodos anexados às interfaces Java. A API SAX define um número de eventos. Porém, pode-se escrever código Java para tratar todos os eventos de interesse da aplicação. Caso a aplicação não necessite de certos eventos, simplesmente esses eventos são desconsiderados, não havendo a necessidade de escrever código algum, pois o analisador os descartará.

Alguns dos eventos SAX são relacionados abaixo:

- startDocument – significa o início do documento.
- EndDocument - significa o fim do documento.
- startElement – significa o início de um elemento. O *parser* dispara este evento quando todas as *Tags* tiverem sido processadas. O que inclui o nome da *Tag* e seus atributos.
- EndElement – significa o fim de um elemento
- Characters – contem o dado, similar ao "Text node" DOM.
- IgnorableWhitespace – pode-se ignorar todos os espaços em branco de um nó apenas com o uso deste evento.
- warning, error, and fatalError – estes três eventos indicam erro na análise. Pode-se respondê-los da forma como quiser.
- SetDocumentLocator – o *parser* envia este evento para permitir o armazenamento do "SAX Locator object", com o qual pode-se encontrar exatamente no documento aonde o evento ocorreu.

3.7.3. Vantagens e Desvantagens da SAX

Uma interface baseada em evento consome menos recursos que uma baseada em objeto, simplesmente porque não precisa criar uma árvore do documento.

A interface baseada em eventos são de níveis mais baixo que as interface baseada em objetos, o que no lado positivo significa mais controle sobre a análise e permite a otimização da aplicação. Já no lado negativo, aumenta o trabalho do programador.

Com SAX a aplicação pode começar o processamento do documento enquanto o *parser* o lê, o mesmo não ocorre com a interface baseada em objeto, visto que a aplicação necessita esperar até que toda a estrutura em árvore tenha sido montada.

SAX é mais eficiente quando não necessita de todo o conteúdo do documento XML, pois a informação armazenada em memória é apenas aquela definida como necessária pela aplicação através do evento e conteúdo.

SAX possibilita a criação de uma árvore do documento lido através do uso da aplicação, a qual deverá colocar explicitamente em um *buffer* os eventos de seu interresse.

O desenvolvedor cria todos os métodos necessários a manipulação da aplicação.

Foram apontadas, no decorrer desse capítulo, as principais características de cada tecnologia e seus pontos fundamentais, demonstrando o relacionamento existente entre elas. A compreensão do exposto é fundamental e necessária para a construção e manipulação dos catálogos eletrônicos, descritos a seguir.

4. Catálogos Eletrônicos

O catálogo eletrônico é um documento (arquivo) o qual contém informações detalhadas sobre produtos ou serviços, oferecidos por entidades comerciais, que proporcionam aos clientes o auxílio à tomada de decisão no momento da compra, propiciando também o desenvolvimento de ordens de pedido e formas de pagamento. Alguns catálogos não possuem artigos para venda, atuando apenas como um índice, organizando informações, fornecendo direcionamento para outros locais, onde um cliente pode encontrar o que deseja. Um catálogo eletrônico não é apenas uma réplica de um catálogo impresso, ele é um novo meio de transmissão de informações, considerado o "portal virtual" ou porta de entrada da empresa estabelecida no mercado global. Catálogos eletrônicos possuem quatro características importantes: interatividade, atualização dinâmica, hipertextualidade e alcance global [SEGEV, WAN & BEAM, 1995].

A seguir, será apresentada uma pequena descrição de cada característica.

4.1. Interatividade

Catálogos impressos representam um fluxo de informação em um só sentido, da companhia para seus clientes. Catálogos eletrônicos provêem um canal de comunicação nos dois sentidos. Por exemplo, se um cliente tem uma dúvida ou comentário sobre um produto observado no catálogo, basta enviar um e–mail e entrar em contato com a empresa. Esse procedimento é instantâneo, proporcionando *feedback* na relação da empresa com o cliente.

4.2. Atualização dinâmica

Os catálogos eletrônicos são armazenados em um servidor da WEB, acessados pelo *browser* do cliente e distribuído ao redor do mundo. O conteúdo do catálogo pode ser facilmente e constantemente atualizado no servidor. Esta natureza dinâmica dos catálogos eletrônicos permite às empresas responderem rapidamente às condições variáveis do mercado.

4.3. Hipertextualidade

Os catálogos eletrônicos proporcionam um modo de unir muitas características relacionadas aos produtos. Por exemplo, um catálogo de vinhos poderia não apenas conter características próprias, como também ligações com o estabelecimento vinícola produtor de um vinho em particular, com receitas feitas com vinho etc. Estas informações podem ser "navegáveis" a critério do usuário.

4.4. Presença global

A Internet possui alcance global. Conseqüentemente, qualquer catálogo eletrônico é um ponto de compra em um mercado global, transcendendo as limitações tradicionais de tempo e espaço. Como o custo para criar catálogos eletrônicos é mínimo comparado às formas tradicionais de catálogos impressos em papel, pequenas e médias empresas tem a chance de competir globalmente.

Através das características definidas acima, catálogos eletrônicos podem ser vistos como um canal interativo responsável por estabelecer relacionamentos entre uma empresa e seus fornecedores, parceiros empresariais e consumidores.

4.5. Características funcionais dos catálogos eletrônicos

Uma vez que catálogos eletrônicos são um novo “portal” para a empresa, eles são usados para servir a vários tipos ou objetivos de negócio, incluindo marketing, vendas, distribuição e suporte ao cliente. Como resultado, um catálogo eletrônico possui variedade funcional. Um simples catálogo pode conter apenas descrições e preços de produtos, não permitindo pesquisa, ou capacidades de pedidos de compra. Em contrapartida, um catálogo completo, integrado ao banco de dados da empresa, pode permitir pesquisa aos usuários, estabelecendo critérios de pesquisa, realizar processamento de pagamentos em tempo real, estar integrado a mecanismos com conexões seguras interligados ao sistema financeiro da empresa.

As características funcionais dos catálogos eletrônicos são divididas em quatro grupos - conteúdo e apresentação, integração, “Computational services” e rastreamento:

- Conteúdo e apresentação: estabelece o item principal a ser armazenado no catálogo: produto, serviço ou ambos. O catálogo eletrônico pode conter informações

detalhadas sobre os produtos através de textos, figuras, gráficos, vídeos ou animações, bem como informar como e onde os produtos podem ser utilizados. Um catálogo eletrônico pode também conter informações oriundas de várias fontes (fabricantes).

- Integração: o catálogo pode estar integrado com o banco de dados da empresa, possibilitando a interação entre diversos sistemas.

- Computational services: o catálogo pode permitir diversas formas de pesquisa de seu conteúdo: navegação, ordenação on–line, geração automática de informações de diversas fontes, cruzando vários bancos de dados. Além disso, o catálogo eletrônico pode estar dotado de mecanismos seguros de pagamento, teste e download (caso o produto seja um software), suporte para e–mail, bate papo, som e vídeo ao vivo, ou forum de discussão.

- Rastreamento: o catálogo pode armazenar informações estatísticas sobre os visitantes do *site* como a quantidade de visitantes e sua localização demográfica, o tempo de permanência, e alterar o conteúdo da informação, dependendo das características do visitante.

4.6. Tipos de catálogos eletrônicos

O catálogo eletrônico é dividido em três tipos principais: standalone, electronic mall, and embedded [SEGEV, WAN & BEAM, 1995].

Um catálogo tipo standalone está localizado em um *site* independente da WEB com a finalidade única de disponibilizar produtos e serviços para venda direta, normalmente sem possibilitar um processo de busca.

Um centro comercial eletrônico (electronic mall) é um conjunto de catálogos pertencente a vários vendedores os quais compartilham o mesmo espaço virtual fornecido por um único provedor Internet. Assim como os catálogos standalone, centros comerciais eletrônicos possuem o propósito de vender produtos ou serviços. Considerando que um centro comercial inclui catálogos de vários fornecedores, o conteúdo dos catálogos tende a variar amplamente.

Catálogos tipo Embedded são os catálogos eletrônicos integrantes de um grande WEB *site*. A diferença principal entre o Embedded e os dois anteriores está no fato daquele não ser utilizado necessariamente para apoiar vendas diretas, e sim para auxiliar outros objetivos da corporação, como marketing, suporte ao cliente e comunicação em relação ao conteúdo, contendo descrições detalhadas sobre produtos ou serviços relacionados (vinculados) a outras empresas.

Catálogos eletrônicos são usados pelas companhias em dois modos primários: vendas a varejo (retailing), com os tipos standalone e electronic mall, e marketing eletrônico, com o tipo Embedded. O fator de sucesso dos catálogos do varejo se concentra no volume de vendas e não no número de visitas diárias, nem no total de visitas mensais. Entretanto, a capacidade de armazenar informações estatísticas é um passo importante para realizar vendas a longo prazo.

Em fevereiro de 1995, foi realizado um estudo com a finalidade de levantar dados suficientes para responder vários questionamentos sobre catálogos eletrônicos, procurando descobrir os tipos existentes na WEB, suas características funcionais, seus pontos em comum, o nível de investimento, os objetivos de negócio e o tipo de empresa que os utilizam, determinando, com isso, valores de medida para avaliar o desempenho dos catálogos. A seguir, é apresentado um resumo dos resultados obtidos [SEGEV, WAN & BEAM, 1995].

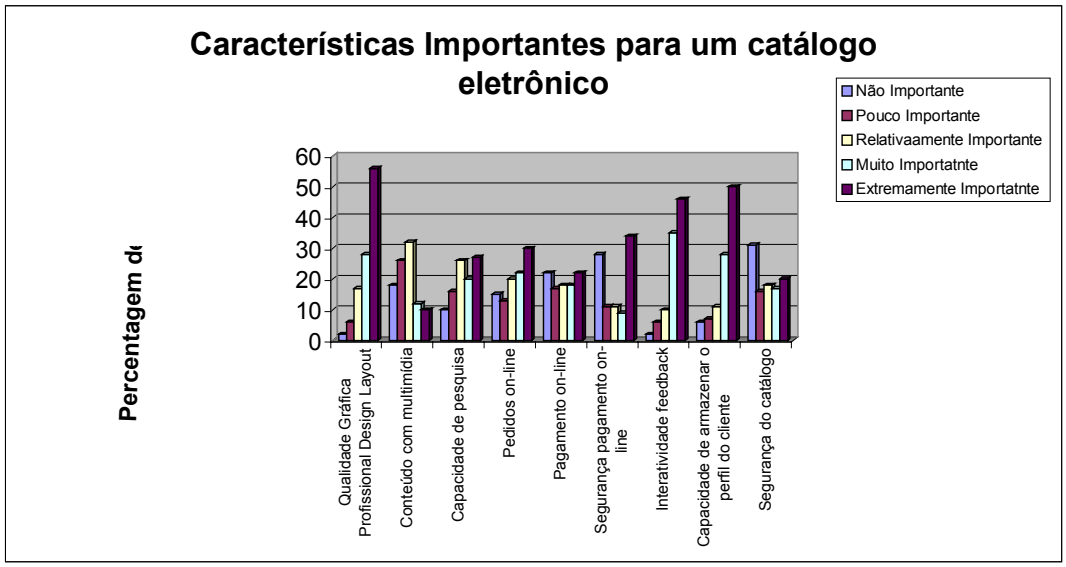
a) Características funcionais mais importantes: Layout e design de ótima qualidade, capacidade de rastreamento, *feedback* e garantia de segurança (veja a fig. 9).

b) Práticas de negócio mais utilizadas: Vendas, marketing, suporte ao consumidor.

c) Fatores de sucesso: quanto maior o investimento nos catálogos (qualidade, pessoal envolvido), maior o número de visitas e, provavelmente, maior volume de vendas.

d) Tipos de catálogos usados: pequenas e médias empresas tendem a utilizar os tipos standalone e eletronic mall, enquanto grandes empresas utilizam, além desses dois, os tipos Embedded.

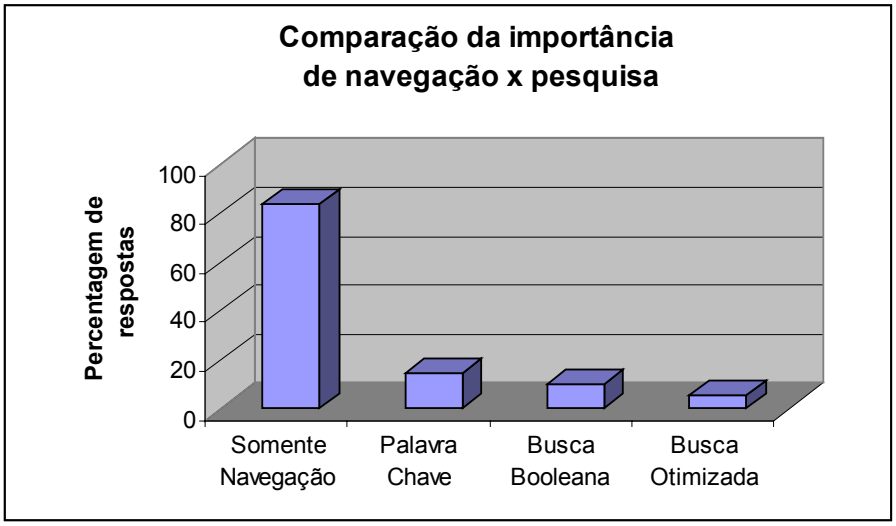
Figura 9 - Características mais importantes para um catálogo eletrônico



Fonte:[SEGEV, WAN & BEAM, 1995].

Curiosamente, a capacidade de busca de informações pelos catálogos foi um item de pequena relevância para as empresas pesquisadas, pois o número de produtos contidos no catálogo era pequeno e o conteúdo desses catálogos era composto por documentos HTML estáticos, gerados dinamicamente através de *queries* em bancos de dados (veja a fig. 10). Além disso, a criação de procedimentos de busca gerava mais custo à aplicação.

Figura 10 - Comparação da importância de navegação X pesquisa.



Fonte: [SEGEV, WAN & BEAM, 1995].

4.7. Catálogos e as novas formas de busca

Qualquer tipo de consumidor tem dificuldade em localizar os produtos desejados na Web, dada a imensa quantidade de páginas de diferentes fornecedores. Para agilizar essa busca, torna–se necessário a padronização de serviços de procura os quais devem encontrar as informações, independentemente do local onde ela se encontra e demonstrá–las num mesmo formato para que o consumidor possa tomar decisões rapidamente (segundo seu critério). Isso facilita a comparação de produtos de diversos fornecedores ao mesmo tempo.

Seja consultando ou comprando, da perspectiva do cliente, o catálogo *on-line* é a parte mais importante do comércio eletrônico, pois contém as informações procuradas pelos clientes: produtos, fotos e preços. Porém, um catálogo web pode ser uma experiência de compra completamente interativa com vídeo, som, e muito mais. Colecionar e incorporar imagens, sons, para centenas (ou milhares) de produtos de catálogo gera um enorme esforço e muito trabalho para organizações.

Os catálogos eletrônicos facilitam a busca de dados de fontes diferentes, possibilitando, a fabricantes e consumidores, localizarem e adquirirem qualquer informação desejada. Proporcionam aos compradores meios de obter rapidamente uma maior quantidade de informação sobre produtos do que por canais convencionais,

aliando a flexibilidade de apresentar esta informação num formato padronizado, facilitando as comparações entre produtos. Além disso, depois da decisão ser feita, os catálogos proporcionam ao participantes da negociação as facilidades oferecidas pelas transações eletrônicas.

4.8. Catálogos customizados

No mundo dos negócios, normalmente as empresas mantém diferenciação entre clientes, promovendo descontos especiais para certos compradores. Os catálogos tornam possível definir preços diferenciados dependendo do tipo do cliente. Geralmente, grandes empresas têm certos descontos especiais no preço dos produtos.

Utilizando–se catálogos customizados é possível reunir informação relevante sobre um produto, mesmo que ele seja produzido por várias empresas, ou seja, o cliente não precisa acessar locais diferentes para obter a informação específica para cada parte de um produto final.

Com essa possibilidade de fornecer informações adicionais sobre produtos externos através de catálogos virtuais em sua própria área, a empresa mantém o cliente "dentro de casa" ao invés de direcioná–lo para outro *site* onde, com certeza, existem links para outras empresas concorrentes.

Antes da criação e utilização de catálogos, alguns pontos devem ser considerados:

- Descrições unificadas dos produtos: até o momento não existe um padrão para a criação de catálogos. Esforços têm sido realizados nesse contexto através de CommerceNet's Interoperable Catalog activities e RossetaNet. Sem tal padronização torna–se difícil e pouco confiável a comparação entre produtos.

- Necessidade de um padrão para comunicação entre catálogos: infelizmente, HTML não é o protocolo certo para esse trabalho, uma vez que foi criado como uma linguagem de apresentação, não como um meio para troca de dados. Com a adoção de um protocolo mais apropriado, os catálogos deverão trocar informação de diversas fontes através de EDI, apesar das muitas diferenças de plataformas, ambientes operacionais, e formatos de dados. Um comitê formado por empresas criadoras de aplicações e empresas de publicação de conteúdo Web, liderado por

Vignette, estabeleceu um protocolo baseado em XML, chamado ICE (Information and Content Exchange).

Para que um catálogo ofereça informações customizadas de seu conteúdo, quatro aspectos devem ser observados [COMMERCENET, 1997]:

- Possibilitar a comparação de produtos semelhantes, oferecidos por fornecedores concorrentes, apresentando as informações num certo formato definido pelo comprador.
- Fornecer grande variedade de produtos oferecidos por vários vendedores.
- Incorporar links que permitam ao comprador obter prontamente informações sobre outros catálogos, descrevendo produtos ou itens de forma a complementar ou adicionar valor ao produto a ser comprando.
- Fornecer uma visão de todos os artigos e serviços pertencentes a um produto particular.

4.9. Catálogos interoperáveis

Diversas interpretações são apresentadas sobre interoperabilidade. De uma forma geral, interoperabilidade significa a habilidade de sistemas separados trabalharem em conjunto, operando como se fossem uma única entidade. Para as transações empresariais de negócio geradoras do comércio na Internet, interoperabilidade significa a habilidade de fazer negócio sem a estipulação e implementação de um sistema proprietário, necessário quando " todo mundo faz de um modo diferente", ou seja, cada empresa tem seu padrão de comunicação, de descrição de produtos etc. Sem interoperabilidade, pode haver perdas de oportunidades, devido aos custos de implantação, necessários aos participantes do negócio. Existem muitos esforços (feitos por entidades como RossetaNet) para facilitar a interoperabilidade em comércio eletrônico, desenvolvendo especificações, normalmente baseadas em XML, para a troca de informação entre companhias, em uma indústria particular ou entre setores.

Com tantas inovações, um negócio na Internet que visa combinar serviços entre vários participantes, criando uma comunidade virtual, pode entrar em conflitos com uma variedade desnorteada de implementações para aplicações de comércio eletrônico.

Esta diversidade é uma barreira crescente à interoperabilidade, elevando os custos de implementação e limitanndo as alternativas para empresas que querem estabelecer novas relações na Internet.

Indústrias dos mais diversos setores precisam criar vocabulários especializados, com termos e propriedades únicas para descrever seus produtos e serviços, mas alguns conceitos necessitam de "especificações verticais", aplicáveis a todos os domínios empresariais. Torna-se imperioso a cada segmento a criação de seus vocabulários, inclusive com relação a definições básicas para um negócio, como padrão para documentos, ordens de compra e faturas, seguramente necessárias para todos os tipos de indústria. Da mesma forma, as especificações industriais contêm, freqüentemente, arquiteturas para a formatação da mensagem e protocolos de comunicação. O Mercado forçará o surgimento de alguma convergência e consolidação em implementação de arquiteturas para comércio eletrônico, como a proposta pela OMG. Mas é impossível predizer quando isso acontecerá. Enquanto isso, novas especificações surgirão. Será muito difícil que um único padrão, ou até mesmo alguns padrões, venha a ser o definitivo por causa da diversidade inerente em modelos de negócio e práticas ao redor do mundo.

Considerando os vários níveis de interoperabilidade, torna–se possível criar uma especificação. Inicialmente, não é necessário para os negociantes concordarem sobre o que eles fazem ou como eles fazem: eles simplesmente têm de concordar com um método comum para descrever o que eles fazem. Além disso, esse método não precisa descrever o aspecto global do sistema. Ao invés, podem ser usados metadados para descrever as características do sistema pertinentes ao potencial da integração. Em outras palavras, pretende–se que este método forneça uma base comum para dois parceiros de negócio, ou seja, uma compreensão de como eles estabelecerão a negociação. A maioria dos detalhes de implementação dos sistemas é irrelevante, e não precisa ser demonstrado, permitindo liberdade aos negócios, para se diferenciar em extensão ou em qualidade da interoperabilidade.

A interoperabilidade universal nem sempre é necessária. Para uma empresa participante da negociação decidir se pode negociar com outra, deve responder um conjunto de perguntas as quais visam definir o nível de interoperabilidade necessário.

Obviamente, os pré– requisitos mais básicos para interoperabilidade se concentram em encontrar um parceiro empresarial em potencial, descobrir quais serviços ele oferece, e só então investigar se pode negociar com ele. Esta investigação envolverá o estudo das interfaces de aplicação, calculando o custo de quaisquer modificações no sistema atual.

Além da idéia de se estabelecer níveis de interoperabilidade, são indicados documentos XML para descrever as interfaces de aplicação. A descrição de interfaces, em termos de documentos determinadores de propriedades, aumenta a transparência da implementação e torna a interoperabilidade mais fácil de ser alcançada e mantida. A grande meta é a criação de empreendimentos virtuais pela combinação de serviços de diferentes negócios. Como exemplo, uma comunidade de negócios, como as empresas participantes da fabricação de automóveis, pode descrever os tipos de documentos que trocam entre si, definindo documentos padrões de negócio. Sendo assim, compradores, fornecedores ou até transportadores, podem se beneficiar do sistema. Isso eleva a integração "do nível do sistema para o nível do negócio".

Com base nas informações acima, pode-se definir um catálogo interoperável como um "mecanismo" facilitador para a implementação de procedimentos entre as organizações envolvidas no negócio, tornando mais eficiente a busca e troca de informações. Além disso, um catálogo deve possuir meios de proporcionar a satisfação das expectativas dos usuários finais. Sendo assim, o catálogo deve proporcionar:

- A habilidade aos interessados em encontrar informações precisas, úteis e adicionais sobre os produtos e serviços desejados;
- A facilidade de encontrar, avaliar e comparar produtos, oferecidos por vários vendedores de uma maneira simples, conveniente e fácil de usar (preferencialmente com um *browser*);
- Possibilitar a um grande número de fornecedores, grandes e pequenos, participar da solução, sem ter de fazer investimentos significantes em tecnologias de catálogos eletrônicos;
- A habilidade de facilitar a publicação eletrônica de ordens de compra e pagamentos, para o fornecedor escolhido;

- A habilidade de possibilitar o isolamento e integridade dos dados, pertencentes às organizações compradoras e vendedoras;
- A possibilidade de armazenar e fornecer a situação (status) do pedido e da entrega;
- A habilidade para produzir relatórios de venda e facilitar a criação de um banco de dados para tomada de decisão.

Além de economizar tempo, dinheiro e esforço, envolvidos na avaliação de produtos e serviços, um sistema gerenciador de catálogos pode ser usado para acomodar rotinas automatizadas que, regularmente, buscam na rede novas informações, ou as atualizam de acordo com o interesse do usuário, "refinando" a informação através de padrões previamente definidos, apresentando os dados mais vitais dentro de um formato personalizado.

Considerando um varejista, vendedor de produtos de múltiplos fabricantes, ele deverá incluir informação do produto de cada fabricante em seu catálogo de produtos. Para armazenar e reproduzir tudo isso em seu catálogo, ocorrerá um considerável custo de manutenção. Para reduzir o volume de informações, uma solução seria, como já mencionado, direcionar o cliente ao *site* do fabricante, de forma que o próprio cliente possa obter informações mais detalhadas sobre o produto. Porém, isso acarreta alguns inconvenientes:

- O cliente pode se perder na página do fabricante, sem saber como voltar ao varejista.
- O fabricante não sabe o contexto das interações do cliente com o varejista.
- O cliente pode entrar em uma página do tipo 'Como fazer o pedido' que o direcione para outro varejista.
- Se o cliente volta para o varejista original usando o botão back, nenhuma informação adicional sobre o produto é levada a seu *site*.

Uma alternativa para esses problemas é criar algum mecanismo, possibilitando aos fabricantes responderem a questões específicas através dos varejistas, para satisfazer as necessidades de informação do produto por parte do cliente. Esta abordagem cria uma relação empresarial entre o varejista e o fabricante, culminando na existência de catálogos virtuais.

A expressão catálogo virtual se deve ao fato desse tipo de catálogo não ser formado por um documento (como um arquivo XML) e não estar armazenado em nenhuma máquina. O catálogo virtual é gerado apenas "em memória", sendo formado pela união de informações advindas de um ou vários catálogos. Por exemplo, o catálogo virtual permite a recuperação de informações de produtos, inseridos no catálogo de um distribuidor, combinando informação dos catálogos de múltiplos fabricantes. Esta recuperação é executada dinamicamente, no pedido do usuário, baseado no critério de procura do usuário. Sendo assim, o catálogo virtual do distribuidor sempre é mantido em dia e em sincronismo com cada catálogo do fabricante. Com catálogos virtuais, o distribuidor mantém controle sobre a interação com o usuário. O usuário nunca interage diretamente com o catálogo do fabricante [Keller, 1997]. A relação entre o usuário, catálogo virtual e catálogos do fabricante é mostrada na figura 11.

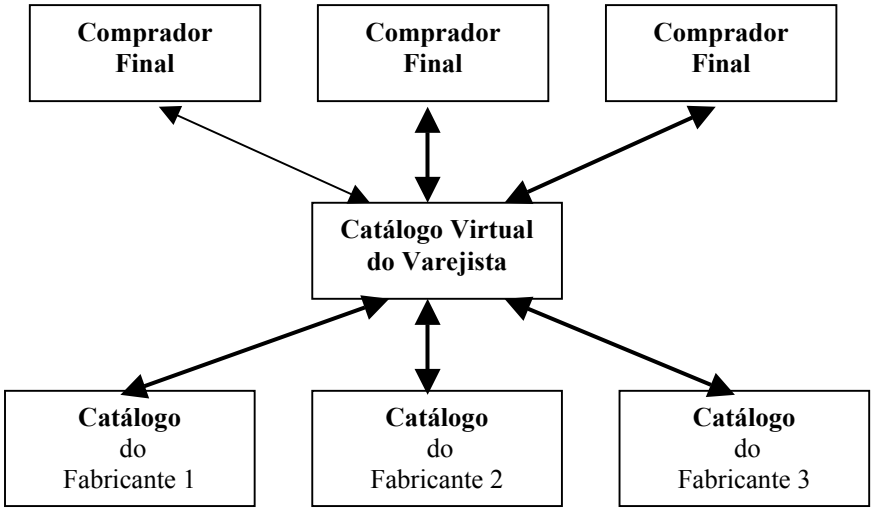


Figura 11 - Interação dos catálogos virtuais criados em XML

4.10. Modelo empresarial básico

Atualmente, quando os compradores procuram produtos na rede, com certeza eles não têm nenhuma outra escolha a não ser buscar informações individuais de cada fabricante, ou seja, um de cada vez. Isso porque cada vendedor deve criar e manter seus catálogos independentemente.

Os compradores seriam mais eficientes com uma infra–estrutura de comunicação a qual lhes permitissem examinar múltiplos catálogos concorrentemente e, então, juntar a informação em qualquer formato por eles especificados.

As vantagens inerentes da divulgação da informação dos produtos através dos catálogos eletrônicos é bastante diferente para compradores e fornecedores, e importante para ambos. Para os fornecedores, os catálogos eletrônicos proporcionam o aumento do alcance da organização, redução de custos de promoção, diminuindo os gastos com produção e distribuição, tipicamente associados aos catálogos impressos.

Para os compradores, incluindo aqui os varejistas, distribuidores e consumidores finais, o catálago eletrônico oferece benefícios dada a maior rapidez e eficiência na escolha de produtos, tornando possível a comparação de produtos de vários fornecedores de uma só vez.

O aparecimento da tecnologia de catálogos não mudará as práticas empresariais por si, porém possibilitará a compradores e fornecedores substituir processos baseados em papel, de difícil controle, por meios equivalentes, tirando proveito das economias inerentes na disseminação de dados através dos meios eletrônicos. Apesar disso, os catálogos eletrônicos não mudarão a relação na cadeia de suprimentos, ou seja, continuarão a existir os fabricantes, varejistas e distribuidores, o que provavelmente mudará será a forma como as empresas divulgarão seus produtos, ao invés de centrar esforços em mala–direta, telemarketing ou através dos catálogos convencionais.

4.11. Benefícios para os envolvidos na adoção dos catálogos

Nesse contexto, todos os envolvidos, sejam fabricantes, distribuidores, varejistas ou consumidores finais, possuem algum tipo de benefício.

Para os fabricantes, uma vez que a descrição dos produtos produzidos estará disponível e interligada aos fornecedores, certamente, outros clientes, visualizando essas informações, poderão interessar–se por algum produto oferecido, aumentando o alcance comercial. Outra vantagem se refere à maior eficiência na manutenção dos catálogos e administração das ordens de pedido, resultando na redução de custos. Para alguns fabricantes, porém, a facilidade dada aos consumidores em comparar as características e preços dos produtos, pode não ser completamente satisfatória. No mínimo, isto provocará um ambiente mais competitivo, no qual as margens de lucro deverão ser revistas.

Para varejistas e distribuidores, podem ser implementados procedimentos automatizados objetivando atualizar a linha de produtos comercializados, através das informações provenientes dos catálogos de cada um de seus próprios fornecedores.

Para consumidores finais, o tempo de procura pelos produtos é reduzido. Além disso, as informações mais completas apoiam a decisão de compra, permitindo identificar claramente as diferenças de preços e características. Devido a atualização automática dos catálogos, os consumidores ficam mais atentos às novidades do mercado.

4.12. Uma visão prática dos catálogos

O único ponto de contato do cliente com o estoque é através do catálogo eletrônico no site da Web. É vital para a viabilidade do negócio que o catálogo seja preciso, atraente, distinto e útil. Um bom site da Web não é simplesmente um item para causar boa impressão, ele é o negócio.

A utilização de catálogos permite, com facilidade, a criação de sumários, índices, produtos ordenados por preço, marca, características, etc. Criar esse catálogo com uma linguagem de marcação, como XML, torna possível gerar todas essas relações a partir de um único documento, evidentemente desde que as informações necessárias para ordenação, indexação das palavras–chave etc., estejam explícitas na marcação.

4.12.1. Publicação de um catálogo XML em um site da Web

Existem basicamente três formas:

I – Processamento do catálogo XML no servidor:

Essa talvez seja a forma mais direta de se publicar o catálogo. Podem ser usadas ferramentas de processamento XML para processá–lo, como um conjunto de páginas HTML interligadas, conforme mostra a fig. 12. Essa abordagem tem a vantagem de ser bastante simples. Para um usuário que esteja navegando num site com esse processo, nenhuma diferença é visualizada, pois no servidor reside um programa responsável por converter o arquivo XML para HTML.

Embora a XML seja relativamente nova, várias organizações têm usado a SGML para gerar automaticamente seus códigos HTML durante anos. Entretanto, algumas

dessas empresas são reservadas quanto a difundir essas informações à comunidade, por proporcionarem vantagem competitiva. Essa abordagem possibilita um alto nível de automação, permitindo a um catálogo de produtos uma atualização rápida em milhares de páginas HTML interligadas. Além disso, como os documentos HTML são gerados a partir de um documento XML, é possível realizar mudanças na forma de apresentação do catálogo (na sua "aparência"), simplesmente modificando o *software* de conversão. Observe–se que a conversão em HTML pode ser realizada durante a execução, assegurando ao usuário a visualização atualizada do catálogo. Apesar das características apontadas, essa abordagem sobrecarrega o servidor e pode reduzir a velocidade de transmissão das páginas HTML, apresentadas ao usuário.

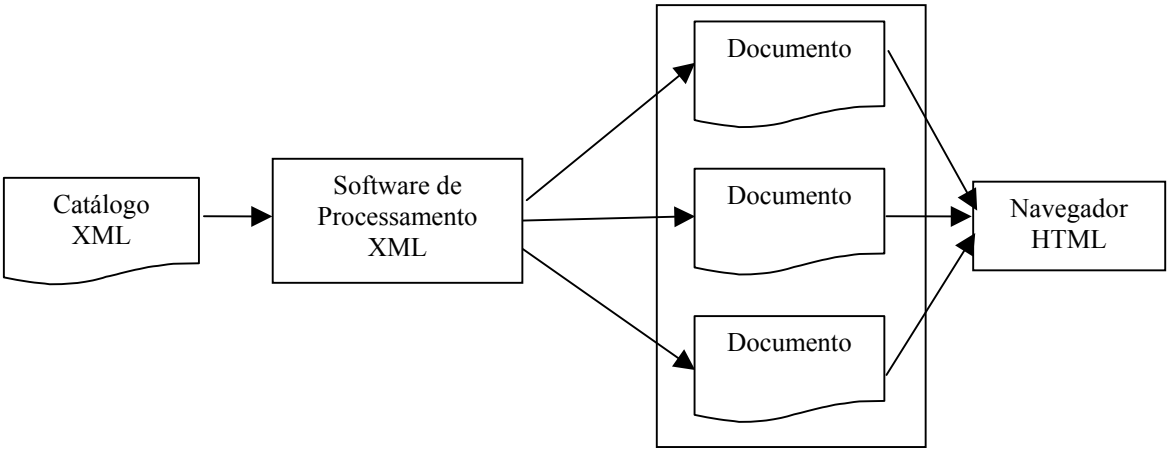


Figura 12 - Conversão de XML em HTML no servidor

II – Processamento do catálogo XML no cliente

Usando essa abordagem, é possível passar a conversão de XML em HTML para o cliente – geralmente, por um *browser* utilizado para navegação na Web. A conversão no cliente pode ser efetuada por um applet. A representação dessa abordagem, pode ser visualizada na fig. 13.

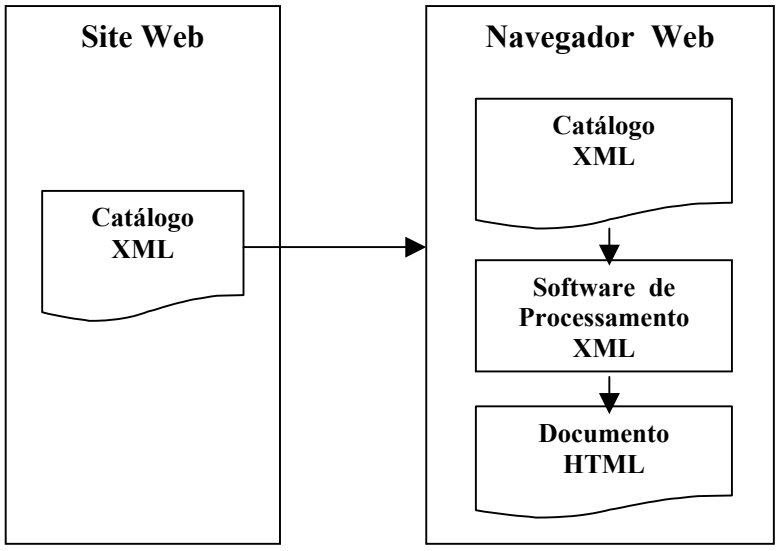


Figura 13 - Conversão de XML em HTML no Cliente

Por exemplo, o navegador Microsoft Internet Explorer versão 4.0 inclui suporte para XML, tanto com applets Java, como com controles ActiveX. Essas ferramentas de programação apresentam uma grande quantidade de opções para processamento e exibição do catálogo no lado do cliente. Entre os recursos possíveis dessa abordagem, destaca–se a geração de tabelas em HTML, ordenadas conforme desejado pelo usuário, a criação de vínculos a páginas de todo o catálogo e um assistente para guiar o usuário através das opções de escolha. Essa abordagem possui a vantagem de não sobrecarregar o servidor, porém o documento XML precisa ser convertido para outro formato. O próximo item demonstra como exibir a XML diretamente, sem necessidade de conversão.

III – Exibindo o catálogo XML diretamente.

Cada vez mais, aparecem navegadores que suportam a exibição direta de documentos XML, evitando assim o problema da necessidade de conversão de XML para HTML. XML é convertida para um formato de exibição agradável, aplicando–se regras de formatação a partir de folhas de estilo XSL. A representação da exibição direta de documentos é vista na figura 14.

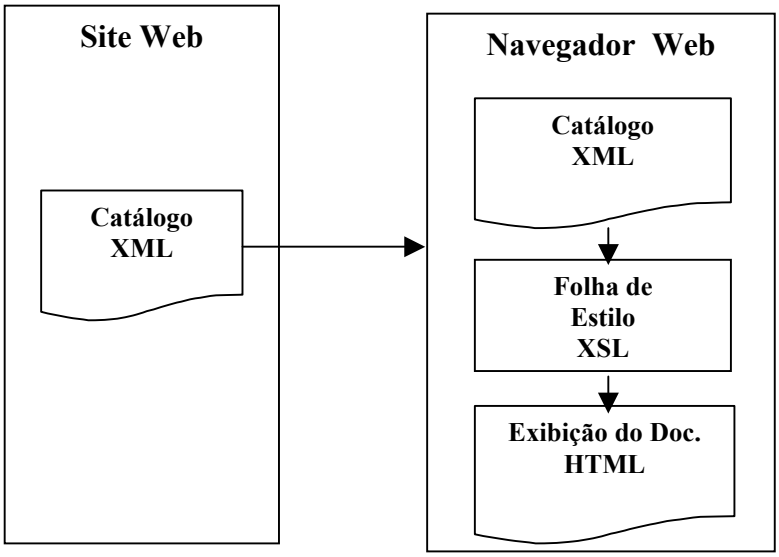


Figura 14 -Representação do catálogo num navegador XML da Web

Essa abordagem, certamente, é a mais inovadora, proporcionando todas as vantagens da utilização das folhas de estilo XSL e demonstrando a formatação e conteúdo do catálogo conforme o desejo do usuário.

4.13. O catálogo como uma ferramenta de vendas

Um outro aspecto a ser observado, ao se divulgar o catálogo pela Web, é a utilização de recursos tecnológicos para obter vantagem competitiva em relação à concorrência. Isso é possibilitado pelo CDF (Channel Definition Format). Este padrão permite estabelecer um canal "push" no site da Web entre os clientes, podendo estes serem informados de lançamentos de novos produtos e ofertas especiais. O conteúdo do canal será uma mistura de comentários sobre vendas e marketing, gerado automaticamente a partir do catálogo. Os clientes podem informar, por exemplo, qual é a faixa de preço interessante, e essas informações podem ser armazenadas num documento em formato XML. No momento em que alguns produtos entrarem em promoção, o cliente pode ser avisado automaticamente. Esse canal de comunicação aumenta o alcance da organização, possibilitando definir o perfil dos clientes, mantendo–os informados das novidades ou promoções (veja a fig. 15).

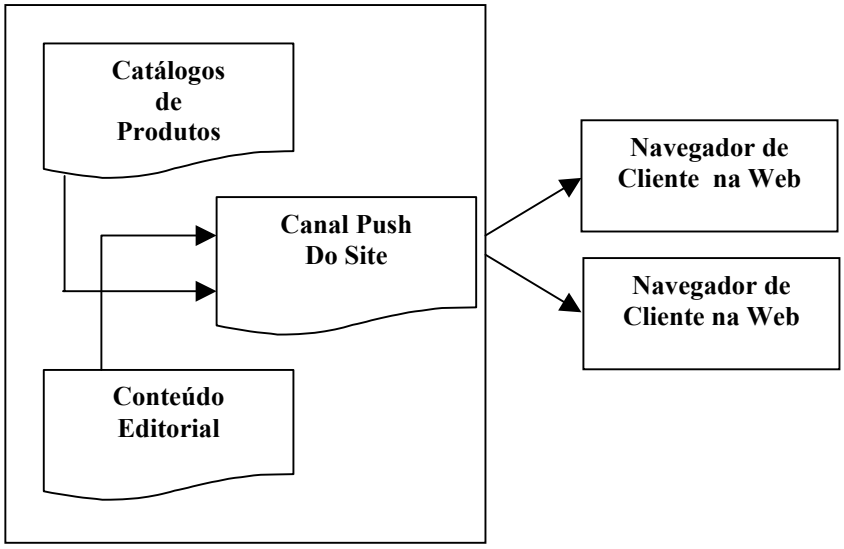


Figura 15 - Canal de comunicação (“push”) com o cliente

4.14. XML Query Language - XQL

XQL – *XML Query Language* é um padrão semelhante a XSL, desenvolvida com o propósito de tornar–se uma poderosa forma de pesquisa em documentos XML. Com as características e possibilidades que XML oferece para a elaboração de catálogos, surge a necessidade de uma linguagem criadora de *queries*, como é o caso de XQL [Texcel, 1998].

XML descreve a estrutura de documentos. A idéia de XQL é tornar o documento XML estruturado em um banco de dados. Isto permitirá poderosas pesquisas para extrair informação da estrutura de XML [Texcel, 1998].

XQL é usada para dirigir e filtrar os elementos e textos contidos em documentos XML. XQL é uma extensão proposta à sintaxe padrão XSL. Fornece uma anotação para apontar para elementos específicos e procurar *nodes* com características específicas. Como exemplo, segue um comando XQL.

Considerando o seguinte trecho de um catálogo em XML:

```
<PRODUTO Fabricante=”Cerâmica Olinda”>
        <NOME>Azulejo</NOME>
        <VALOR>4.54</VALOR>
        <QUANTIDADE>2544 Metros</QUANTIDADE>
</PRODUTO>
<PRODUTO Fabricante=”Cerâmica Bororo”>
        <NOME>Piso</NOME>
```

```
        <VALOR>8.94</VALOR>
        <QUANTIDADE>2240 Metros</QUANTIDADE>
</PRODUTO>
<PRODUTO Fabricante=”Cerâmica Boa Pedra”>
        <NOME>Revestimento</NOME>
        <VALOR>3.56</VALOR>
        <QUANTIDADE>4220 Metros</QUANTIDADE>
</PRODUTO>
```

Uma pesquisa pode ser dada pelo comando: //produto[@fabricante='SAPORE']. Isso quer dizer: mostre todos os produtos do catálogo que possuem SAPORE como fabricante. Então, o resultado da pesquisa será:

```
<xql:result>
<PRODUTO Fabricante=”Cerâmica Olinda”>
        <NOME>Azulejo</NOME>
        <VALOR>4.54</VALOR>
        <QUANTIDADE>2544 Metros</QUANTIDADE>
</PRODUTO>
<xql:result>
```

Esse exemplo demonstra que XQL pode ser usada para pesquisar o conteúdo de catálogos armazenados na rede Internet, proporcionando interoperabilidade entre bancos de dados diferentes.

Utilizando–se uma linguagem de pesquisa, a informação contida na rede pode ser acessível em uma forma estruturada, padronizada e mais eficiente. Além disso, uma variedade de aplicações pode reutilizar esta nova informação gerada. Uma linguagem que selecione dados, segundo um determinado critério, pode apoiar uma variedade de tarefas de processamento de dados nos documentos XML, como examinar, integrar e transformar seus dados. [Robie, 1998].

As características mais importantes de XQL são:

- Um documento XML pode ser examinado como um banco de dados.
- Possibilita pesquisas poderosas expressas de uma maneira bastante simples.
- Utiliza uma sintaxe que permite fazer pesquisas em URL ou embutí–las em atributos do próprio documento XML.

No decorrer desse capítulo, foi demonstrado os tipos de catálogos existentes e suas funções principais, procurando enfocar as razões de sua existência. Para o presente trabalho, a função principal se refere à pesquisa em catálogos, cuja abordagem será tratada no capítulo 6 através da descrição de uma implementação que mostra, na prática, os conceitos estudados.

5. Java

Java apareceu como parte de um projeto de investigação para desenvolver *software* avançado visando uma grande variedade de dispositivos interligados em rede. O objetivo fundamental era criar um ambiente de trabalho pequeno, eficiente, confiável e que funcionasse em tempo real.

As dificuldades tidas com a linguagem original do projeto, C++, levaram à preferência pela definição de uma linguagem inteiramente nova. A linguagem Java foi assim criada, unindo conceitos extraídos de linguagens orientadas a objetos como C++, Eiffel, SmallTalk.

Java foi desenvolvida para ser uma linguagem robusta, independente de plataforma ou sistema operacional, totalmente orientada a objeto, e se popularizou com pequenos aplicativos chamados *applets*. Tais aplicativos são carregados, embutidos em páginas HTML, e rodam geralmente dentro dos *browsers* que acessam a internet. Devido às suas características, cada vez mais ela vem sendo utilizada para desenvolver aplicações executadas *stand alone* independentes da internet.

Para descrever as características de Java, usar-se-á os mesmos adjetivos tecnológicos que a Sun Microsystems empregou em um dos documentos de apresentação da linguagem:

5.1. Orientada a objetos

Java é uma linguagem de programação orientada a objetos. Do ponto de vista do programador, isso significa que o programa é construído em função dos dados a serem manipulados e dos métodos manipuladores desses dados. Juntos, os dados e os métodos que os manipulam, procuram **simular o comportamento dos objetos do mundo real**. Daí o nome **orientação a objetos**.

Em um sistema orientado a objetos, uma classe é uma coleção a qual contém, ou **encapsula**, os dados e os métodos que operam sobre esses dados. Conjuntamente, os dados e os métodos descrevem o estado e o comportamento de um objeto.

As classes são arranjadas em **hierarquias**, nas quais uma **subclasse** pode herdar as características de sua **superclasse**. Uma hierarquia de classes sempre tem uma classe raíz, ou seja, a classe que contém o comportamento mais genérico.

5.2. Interpretada

A explicação para a independência de plataforma de Java reside no fato de ser uma linguagem **interpretada**: o compilador Java gera um tipo de código especial, chamado ***bytecode***, executado por uma máquina virtual implementada em *software*, chamada **JVM** (*Java Virtual Machine* – Máquina Virtual Java). Com C++ é diferente: um compilador C++ gera código para uma máquina específica, por exemplo, um microcomputador e, por conseqüência, o programa somente pode rodar nessa máquina alvo, conforme mostra a tabela 4:

Linguagem tradicional		
Código fonte	**Compilador** (traduz para código executável)	**Código de máquina**
Escrito em uma linguagem tradicional como C++, Basic, Pascal	Para PC	Este código de máquina só roda em PCs
	Para Macintosh	Este código de máquina só roda em Macintosh
	Para workstation Unix	Este código de máquina só roda em workstations Unix
	Etc.	Etc.

Tabela 4 – Relação compilador x código fonte da linguagem tradicional

<table>
<tr><td colspan="4">Linguagem Java</td></tr>
<tr><td>Código fonte</td><td>Compilador Java</td><td>JVM (Java Virtual Machine - Máquina Virtual Java)</td><td>bytecode Java</td></tr>
<tr><td rowspan="4">Escrito em linguagem Java</td><td rowspan="4">Traduz para bytecode, que roda na JVM</td><td rowspan="4">A mesma JVM é implementada em PCs, Macintosh, máquinas Unix e qualquer outro tipo de computador</td><td>O bytecode roda na JVM do PC</td></tr>
<tr><td>O mesmo bytecode roda na JVM do Macintosh</td></tr>
<tr><td>O mesmo bytecode roda na JVM de uma workstation Unix</td></tr>
<tr><td>Etc.</td></tr>
</table>

Tabela 5 - Relação compilador x código fonte da linguagem tradicional Java

Já na tabela 5, como visto, para rodar um programa Java, é usado o interpretador Java (contido na JVM) para executar o *bytecode* resultante da compilação. Como o *bytecode* Java é independente de plataforma, os programas Java podem rodar em qualquer plataforma para a qual a JVM tenha sido portada, posto que a mesma inclui o interpretador mais o sistema de *run-time*.

Em um ambiente interpretado, a fase de **linkagem** existente no desenvolvimento de programas tradicionais compilados praticamente desaparece. O equivalente em Java à fase de *link* consiste apenas no processo de carregar novas classes no ambiente de execução da JVM, processo leve e incremental, que ocorre em tempo de execução, diferentemente do ciclo *compile-link-run*, mais trabalhoso, comum em linguagens como C e C++. O resultado é uma redução no tempo de desenvolvimento dos programas.

5.3. Portabilidade

A portabilidade de Java está ligada ao fato da linguagem ser interpretada e definir um formato de *bytecode* padronizado e neutro em relação às arquiteturas, somado à garantia de não existerem aspectos "dependentes de implementação" na especificação da linguagem. Por exemplo, Java especifica explicitamente o tamanho de cada um dos tipos de dados primitivos, bem como seu comportamento aritmético, o que não acontece

em C, por exemplo, onde um tipo int pode ter comprimento de 16, 32 ou 64 bits, dependendo da plataforma.

Embora seja tecnicamente possível escrever programas não-portáveis em Java, é relativamente fácil evitar as poucas dependências de plataforma expostas pela API Java, escrevendo programas verdadeiramente portáveis, ou "Java puro". O programa "100% Pure Java" da Sun ajuda os desenvolvedores a terem certeza (e obterem certificação) de que seu código é portável.

Há cuidados necessários e relativamente simples visando evitar armadilhas que tornam os programas não-portáveis. Ou seja, realmente é possível seguir o lema, uma das marcas registradas de Java: "Escreva uma vez, rode em qualquer lugar" ("Write once, run anywhere").

5.4. Dinâmica e distribuída

Java é uma linguagem **dinâmica**: qualquer classe Java pode ser carregada em um interpretador Java a qualquer momento, mesmo quando ele já está rodando. Essas classes carregadas dinamicamente podem, então, ser instanciadas dinamicamente. Bibliotecas de código nativo também podem ser carregadas dinamicamente.

Em Java, a orientação a objetos é tão completa que existe até mesmo uma classe para representar as classes, chamada, obviamente, *Class*. Inicialmente, isso pode parecer confuso e excessivamente abstrato. O fato é que, usando a classe *Class*, é possível obter dinamicamente informações sobre uma classe em tempo de execução. Essa funcionalidade é reforçada em Java 1.1, com o acréscimo da API **Reflection**.

Há de se ressaltar também o caráter **distribuído** de Java, pois proporciona considerável suporte de alto nível à utilização em redes. Por exemplo, a classe URL, e outras relacionadas, contidas no package java.net, torna quase tão fácil ler um arquivo ou outro recurso remoto quanto ler um arquivo local.

Em Java 1.1, a API **RMI** (*Remote Method Invocation* – Invocação de Métodos Remotos) permite a dado programa Java chamar métodos de objetos remotos, como se fossem objetos locais.

Java oferece, ainda, o suporte a redes tradicionais, de baixo nível, para quando tal suporte for necessário, incluindo datagramas e conexões através de *sockets* baseadas em fluxos (*streams*).

A natureza distribuída de Java, combinada com sua capacidade para o carregamento dinâmico de classes, representa um recurso extremamente poderoso, tornando possível ao interpretador Java baixar (*download*) códigos da Internet, e executá-los. Conforme se verá adiante, Java implementa fortes medidas de segurança para proporcionar a certeza de que tal procedimento somente será realizado de forma segura. Um exemplo dessa funcionalidade acontece quando um *Web browser* baixa um *applet* Java e o executa.

E a flexibilidade não pára aí, podendo ser usada para resolver problemas muito mais complexos. Como exemplo, reporte-se a um programa multimídia escrito em Java. Quando se pede ao programa para exibir algum tipo de dado nunca encontrado antes, Java torna possível baixar dinamicamente da rede classes capazes de lidar com esse tipo de dado, e exibir os dados dentro de um documento composto. Essas classes podem estar implementadas na forma de componentes JavaBeans. Portanto, o programa Java é capaz de utilizar recursos distribuídos da rede para crescer dinamicamente, adaptando-se às necessidades do usuário.

5.5. Segura

Talvez pelo fato de Java estar fortemente associado à Internet, um dos aspectos mais divulgados da linguagem é sua **segurança**, o que é natural: sem a certeza da segurança, provavelmente não se faria baixar códigos de um *site* desconhecido na Internet, deixando-o rodar no próprio computador. E isso já tem sido feito todos os dias, com os **applets** Java.

Java foi projetado tendo a segurança como característica sempre presente, oferecendo várias camadas de controles de segurança os quais protegem contra código malicioso e permitem aos usuários rodarem tranqüilamente programas de origem desconhecida, como os *applets*.

No nível mais baixo, a segurança é uma conseqüência da robustez de Java, pois os programas Java não podem forjar ponteiros para a memória, nem estourar *arrays*, nem

tampouco, ler memória que esteja fora das fronteiras de um *array* ou *string*. Esses recursos são uma das principais defesas de Java contra código malicioso: ao impedir totalmente qualquer acesso direto à memória, toda uma enorme classe de ataques à segurança é evitada.

Associada à proteção da memória está o processo de verificação do *bytecode*, executado pelo interpretador Java sobre qualquer código de origem desconhecida que é carregado. Essa verificação assegura ao código uma boa formação – isto é, impede que ele avance na memória acima ou abaixo dos limites da pilha (*stack overflow* ou *underflow*), ou implemente *bytecode* ilegal. Se esse passo de verificação do *bytecode* não existisse, o código corrompido por incompetência ou má fé poderia tirar partido de pontos fracos na implementação de um interpretador Java.

Outra camada de proteção para segurança é comumente chamada de modelo de **caixa de areia** (**sandbox**): o código de origem desconhecida pode rodar em segurança, já que mantido isolado dentro de uma caixa de areia, sem causar qualquer dano ao "mundo real", que é o ambiente Java como um todo.

Quando um *applet*, ou outro código de origem desconhecida, está rodando dentro da caixa de areia, fica submetido a diversas restrições de execução. A mais óbvia consiste em o *applet* não ter acesso, de nenhum tipo, ao sistema de arquivos local. Na verdade, existe uma classe, chamada SecurityManager, especialmente para cuidar da implementação dessas restrições. Para assegurar o funcionamento do modelo de segurança, todas as classes do núcleo Java responsáveis por executar operações de risco, como acesso ao sistema de arquivos, primeiro pedem permissão ao SecurityManager atualmente instalado e, se a chamada está sendo feita, direta ou indiretamente, por código de origem desconhecida, o gerenciador de segurança lança uma exceção, impedindo a operação.

A versão Java 1.1 implementa um recurso adicional à questão da segurança: a **assinatura digital**. Anexando uma assinatura digital ao código Java, a origem desse código pode ser estabelecida de forma criptograficamente segura e impossível de falsificar.

Dessa forma, o usuário pode definir que uma determinada pessoa ou organização merece sua confiança. Por conseqüência, o código que traz a assinatura digital dessa

entidade merece confiança, mesmo sendo carregado através da rede, podendo rodar sem as restrições do modelo de caixa de areia.

Porém, quando se trata de segurança, é preciso ser realista. Do mesmo modo que nunca se pode garantir um programa 100% livre de *bugs*, nenhuma linguagem ou ambiente tem a garantia de ser 100% seguro. Dentro desses limites, Java com certeza oferece um bom nível de segurança para a maioria das aplicações práticas, já que antecipa e se defende contra a maioria das técnicas as quais têm sido usadas para fazer com que *software* tenha comportamento malicioso.

5.6. *Multithreaded*

Multithreaded significa um aplicativo com **várias linhas de execução** rodando em paralelo ao mesmo tempo. Para entender esse conceito, imagine-se um Web browser, como o Netscape Navigator ou o Microsoft Internet Explorer, no qual há vários processos acontecendo ao mesmo tempo.

Java é uma linguagem *multithreaded*, ou seja, facilita a criação de aplicativos *multithreaded*, característica intrínseca a essa linguagem. Por exemplo, o coletor de lixo é uma linha de execução sempre rodando, na retaguarda.

Um importante benefício das múltiplas linhas de execução é a melhora da *performance* interativa percebida pelo usuário em aplicativos gráficos.

Enquanto os programadores C ou C++, os quais trabalham com *threads* nessas linguagens, têm grandes dificuldades; em Java, o uso de *threads* é facilitado devido ao fato de a linguagem ter, originalmente, todo o suporte necessário para a criação e o controle de *threads*. A classe *Thread*, do *package* java.lang, contém métodos para iniciar/parar *threads* e definir suas prioridades, entre outras vantagens.

Além disso, a própria sintaxe da linguagem oferece suporte direto para *threads*, através da palavra-chave *synchronized*. Com essa palavra-chave, simplifica-se a determinação de seções de código, ou métodos inteiros, os quais somente devem ser executados por único *thread* de cada vez.

Ou seja, se *threads* são algo que só os gurus conseguem usar em C e C++, seu uso é lugar comum e até obrigatório em Java. Na verdade, muitas das classes da biblioteca

original de Java exigem seu uso em várias situações. Por exemplo, um *applet* que execute uma animação, o faz usando uma *thread*.

5.7. Servlets

Servlets são aplicativos escritos em Java, os quais podem ser acoplados em diversos tipos de servidores para expandir as suas funcionalidades. Divergem dos *applets* por não possuírem uma interface gráfica, e poderem ser escritos para trabalhar com diversos tipos de protocolo de comunicação, como SMTP, HTTP e IRC.

5.7.1. Estrutura básica de um servlet

Para criar um *servlet*, utiliza-se a API *javax.servlet* encarregada de definir a interface *Servlet*. Todo *servlet* deve ser uma implementação dessa interface, ou uma subclasse de uma classe que a tenha implementado. A API não pressupõe nenhum requisito quanto à maneira como os *servlets* serão carregados, ou o ambiente no qual eles serão executados, nem o protocolo a ser utilizado na troca de informações. Nela estão definidos todos os métodos para estabelecer a comunicação com clientes. Um exemplo de classe a qual implementa a interface *Servlet* é a HttpServlet, utilizada para desenvolver *servlets* instalados em servidores Web e que utilizam o protocolo HTTP para enviar e receber informações.

A comunicação entre um *servlet* e um cliente se dá por meio de dois objetos instanciados das classes: ServletRequest e ServletResponse. O primeiro encapsula as funções de comunicação do cliente para o servidor, permitindo ao *servlet* receber dados, como o conteúdo de um formulário HTML. O segundo encapsula as funções de comunicação do servidor para o cliente. Um exemplo seria enviar HTML em resposta aos dados recebidos.

A estrutura mínima de um *servlet* é composta por três métodos da interface Servlet da API, os quais devem ser implementados: *init, service* e *destroy*.

O *init* é o primeiro método a ser executado no momento em que o servlet está sendo carregado. Responsável pela leitura de configurações estabelecidas pelo sistema, pela inicialização de estruturas de dados e, se for o caso, pela realização de conexão com uma base de dados a ser utilizada pelo *servlet* durante a sua execução. Se o *servlet* necessita de algum recurso não disponível no sistema no momento da inicialização, o

init deve lançar uma *UnavailableException*, método não executado novamente, a menos que o servlet seja recarregado. Para recarregar, é necessário remover o *servlet* invocando o método *destroy*, no qual serão executados os procedimentos para a finalização segura do *servlet*.

No processo de inicialização dos *servlets,* o servidor pode estabelecer parâmetros como, por exemplo, nome da base de dados ou diretórios de arquivos temporários. A forma como tais parâmetros são armazenados é específica para cada servidor. Apesar disso, a forma do *servlet* tratá-los será sempre a mesma. O método *init* recebe como argumento um objeto ServletConfig, utilizado para obter os parâmetros de inicialização. A maneira mais simples de realizar essa operação é invocar o método da classe pai, super.init, para lidar com esse objeto. Para entender melhor, observe-se o código abaixo:

```
public void init(ServletConfig configuracao) throws ServletException {
super.init(configuracao);
String parametro01 = getInitParameter("Parametro01");
// Se o parâmetro não esta disponível, ocorre uma Exception
if (parametro01 == null){
throw new UnavailableException (this,
"Parâmetro01 não esta disponível");
}
}
```

Após o término da inicialização, o *servlet* está apto a receber e processar requisições de acesso de clientes. O processamento dos pedidos ocorrem no método *service*, responsável por receber informações do cliente, realizar o processamento, e enviar a resposta.

A cada acesso, o *servlet* cria um *thread* para executar o método *service,* processo o qual exige alguns simples cuidados, conforme já citado, garantindo que o servlet será executado corretamente com vários *threads* simultâneos. Por exemplo, se o cliente irá atualizar uma variável compartilhada, tal variável deve ser sincronizada para assegurar a não ocorrerência de interferência entre *threads*.

Se o método *service* não deve ser executado em vários *threads*, a única alteração necessária no código é definir o *servlet* como uma implementação da interface SingleThreadModel, a qual permite ao método *service* do *servlet* ser executado por um único *thread* de cada vez.

Ao remover um *servlet* do servidor, o método *destroy* é executado. Esse método deve ser utilizado para fazer operações de 'limpeza' das estruturas de dados, para gravar fisicamente o estado do *servlet*, ou para desalocar outros recursos do sistema que estejam sendo utilizados. No momento em que o *servlet* está sendo removido, vários *threads* desse *servlet* podem estar sendo executados, compartilhando recursos, como conexão a rede ou a bancos de dados. Para evitar que tais recursos sejam desalocados enquanto ainda são necessários, os acessos a eles devem ser sincronizados. Esse método, assim como o *init*, será executado uma única vez, e não poderá ser executado novamente a menos que o *servlet* seja reinicializado.

5.7.2. Vantagens dos servlets

Uma vez inicializado, o *servlet* estará apto a lidar com centenas de acessos simultaneamente, disparando, para cada acesso, um novo *thread*. Os *threads* de um mesmo aplicativo utilizam um espaço de endereçamento de memória comum a todos, permitindo o compartilhamento de dados e recursos do sistema. Ou seja, todos os *threads* de um *servlet* podem fazer uso de uma única conexão já estabelecida com um banco de dados no momento de inicialização do *servlet*. A conexão permanecerá aberta até que o *servlet* seja desativado, saia da memória, ou seja recarregado.

Por serem escritos em Java, linguagem projetada para ser independente de plataforma, conforme já foi mencionado, é possível escrever um *servlet* para um servidor UNIX, o qual poderá ser instalado em um Windows NT, sem a necessidade de rescrever o código, ou recompilá-lo.

A orientação a objetos, característica do Java, favorece a modularização dos aplicativos, tornando-os mais escalonáveis. Como exemplo a autenticação de clientes. Em uma estrutura montada em três camadas: uma primeira, chamada *gateway*, a qual executa um *servlet* responsável por autenticar os usuários e redirecionar os acessos para a segunda camada; uma segunda, composta por servidores que executam *servlets* compostos pela aplicação propriamente dita; uma terceira, formada por servidores de banco de dados. Considerando ser o processo de autenticação bem mais leve que a aplicação, um único *gateway* seria capaz de atender a vários servidores de aplicação. Se, em determinado momento, for detectada sobrecarga dos servidores de aplicação, bastará o acréscimo de mais um servidor à segunda camada, e a informação ao servidor do

gateway sobre a existência de mais um servidor capaz de receber o redirecionamento de acessos de clientes autenticados.

Os *servlets* possuem outra característica, qual seja a capacidade de acesso por meio de *tags* <SERVLET> e </SERVLET> inseridos no código HTML. Apesar de não ser suportada por todos os servidores aptos a executarem *servlets*, *tags* podem ser bastante útéis para aplicativos simples, como ao contador de acessos.

5.8. *Commom gateway Interface* - CGI

Programa capaz de ser escrito em diversas linguagens, como PERL ou C, o qual se comunica com servidores Web através de interface padronizada, conhecida como CGI, *Commom Gateway Interfacce.* Esses aplicativos são chamados genericamente de CGIs.

Cada vez que um CGI é executado, é necessário carregá-lo na memória e iniciar o processo, o qual será extinto com o término da execução do aplicativo. Caso seja uma linguagem interpretada, como *PERL*, soma-se o tempo necessário para compilá-lo. Como são processos distintos, existe uma grande dificuldade no compartilhamento de recursos do sistema, por exemplo, conexão a banco de dados, como também na troca de informações entre os processos executados simultaneamente.

6. Desenvolvimento da aplicação de busca em documento XML

A criação de catálogos escritos em XML virá contribuir para o surgimento de serviços diferenciados nos mais diversos setores da economia. Serviços especializados são realidades hoje, e com a adoção da XML, os mesmos tornar-se-ão mais fáceis de serem implementados e mais eficientes, como a busca por produtos e seus revendedores ou fabricantes, a cotação de preço para produtos, as formas de pagamentos.

A seguir, serão descritos os procedimentos necessários para a criação de uma aplicação, utilizando recursos da XML, a fim de propiciar o desenvolvimento de um serviço de busca otimizada, o qual permita ao usuário obter o melhor preço de um bem ou serviço dentre os *sites* cadastrados em determinado prestador de serviço.

6.1. Elaboração do catálogo

A aplicação desenvolvida é voltada ao segmento de informática, sendo elaborado catálogo com aproximadamente 30 itens, incluindo microcomputadores, impressoras, acessórios, *softwares* diversos.

É necessário, nesse momento, recordar dois aspectos fundamentais já abordados:

I – A linguagem XML permite a criação de qualquer tipo de *Tags*, dependendo das necessidades de armazenamento do catálogo.

II – A fim de possuir, a consulta realizada, resultado correto e confiável, os produtos devem ser descritos da mesma forma nos catálogos dos fornecedores cadastrados, ou seja, deve ser criada uma padronização (das *Tags*) com relação à codificação dos produtos, de maneira a fornecedores identificarem determinado produto da mesma forma. Para melhor esclarecimento desse aspecto, a seguir é apresentado um trecho do catálogo de um dos fornecedores.

```
<COMERCIOELETRONICO NOME="ONE">
        <PRODUTO>
                <CATEGORIA>Hardware</CATEGORIA>
                <TIPO>Microcomputador</TIPO>
                <FABRICANTE>Acer</FABRICANTE>
                <NOME>Power 6100 Pentium III</NOME>
                <DESCRICAO>Pentium III 500Mhz, 64MB RAM, HD 10GB, CD ROM 40x, Placa de
Vídeo 4Mb AGP, FAX/MODEM 56K, 1 ano de garantia</DESCRICAO>
                <VALOR>2.699,00</VALOR>
                <QUANTIDADE>15 unidades</QUANTIDADE>
```

```
            <PAGAMENTO>Cartao Dinheiro</PAGAMENTO>
        </PRODUTO>
        <PRODUTO>
            <CATEGORIA>Hardware</CATEGORIA>
            <TIPO>Microcomputador</TIPO>
            <FABRICANTE>Compaq</FABRICANTE>
            <NOME>Presario 5BW143 Pentium III</NOME>
            <DESCRICAO>Pentium III 650Mhz, 64MB RAM, HD 20GB, CD-RW,
FAX/MODEM 56K V90, Camerara Web, Placa Ethernet, 1 ano de garantia</DESCRICAO>
            <VALOR>2.999,00</VALOR>
            <QUANTIDADE>10 unidades</QUANTIDADE>
            <PAGAMENTO>Cartao</PAGAMENTO>
        </PRODUTO>
        <PRODUTO>
            <CATEGORIA>Software</CATEGORIA>
            <TIPO>Aplicativo</TIPO>
            <FABRICANTE>Microsoft</FABRICANTE>
            <NOME>MS-Office</NOME>
            <DESCRICAO>Software para automação de escritório - Editor de texto, Planilha
Eletrônica, Banco de Dados, Apresentação</DESCRICAO>
            <VALOR>790,00</VALOR>
            <QUANTIDADE>10 unidades</QUANTIDADE>
            <PAGAMENTO>Cartao</PAGAMENTO>
        </PRODUTO>
</COMERCIOELETRONICO>
```

Toda padronização utilizada para a criação dos catálogos (*Tags* permitidas, conteúdo permitido para cada *Tag,* sua localização) foi definida num DTD, cujo código é apresentado nos anexos.

A estruturação do DTD foi montada para reduzir o tamanho do código. Se fosse criado *Tag* para cada produto disponibilizado no catálogo, ter-se-ia um DTD grande: um DTD para 50 itens ocupa, em média, um tamanho de 3,35 KBytes, e provoca lentidão desnecessária na rede. Em contraponto, a criação de *Tag* para cada produto aumenta a *performance* de localização das informações.

Decidiu-se, então, pelo desenvolvimento de padrão comum a todos os produtos, reduzindo o tamanho do DTD para 568 Bytes, e aperfeiçoando o mecanismo de busca por informações.

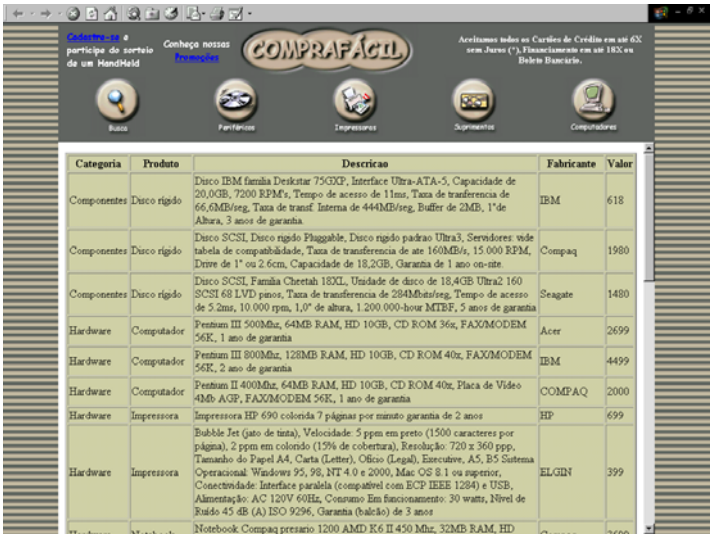
Foram descritas, nos capítulos anteriores, as principais tecnologias empregadas na elaboração e manipulação dos catálogos eletrônicos. O catálogo é um documento XML no formato texto, podendo ser criado a partir de qualquer editor de texto, como o bloco de notas do Windows, porquanto existem ferramentas mais apropriadas que auxiliam na sua construção. Durante a elaboração da aplicação, foram usadas algumas ferramentas

(editores) voltadas à geração de documentos XML: Microsoft XML Notepad e XML Spy, mais especificamente o XML Spy 3.0, por ser uma ferramenta mais completa.

Cada catálogo criado em XML pode ser representado da forma desejada pelos seus fornecedores, utilizando–se, para isso, as características abordadas da XLS no item 3.4.3. A Figura 16 e 17 mostram os catálogos dos fornecedores que, apesar de conterem a mesma estrutura, aparecem com diferentes formatações.

Categoria	Produto	Descricao	Fabricante	Forma de Pagamento	Valor
Software	Antivirus	Ambiente Windows 95 e 98, Idioma Inglês	McAfeeHP	Dinheiro Cartão	90
Software	Antivirus	Ambiente Windows 95 / 98 / NT / 2000, Idioma Português	Norton	Dinheiro Cartão	59
Software	Banco de Dados	Ambiente Windows 95 / 98 / NT, Português	Microsoft	Dinheiro Cartão	636
Software	Banco de Dados	10 usuários, Server: Windows NT 4 e 2000, NetWare 4.2 e 5x, Linux (Red Hat 5.2/6.0, Caldera 2.2 e SuSE 6.1) e Sun Solaris 2.6/7.0 - Clients: DOS e Windows 95/98/NT/2000. - Workgroup/Workstation: Windows 95/98/NT/2000 , Inglês	Stern Software	Dinheiro Cartão	3268
Software	Banco de Dados	Ambiente Windows NT, Solaris, Unix, Inglês / Português, 5 usuários	Oracle	Dinheiro Cartão	1576
Hardware	Computador	AMD 300Mhz, 32MB RAM, HD 4GB, CD ROM 24x, FAX/MODEM 36K, sem garantia	BLASTER	Dinheiro Cheque Cartão	1500
Hardware	Computador	Pentium II 450Mhz, 320MB RAM, HD 17GB, DVD, PLACA DE REDE 1 ano de garantia	Microtec	Dinheiro Cheque Cartão	4000
Hardware	Computador	Pentium II 400Mhz, 64MB RAM, HD 10GB, CD ROM 40x, Placa de V_o 4Mb AGP, FAX/MODEM 56K, 1 ano de garantia	COMPAQ	Dinheiro	2000
Componentes	Disco rígido	Disco IBM familia Deskstar 75GXP, Interface Ultra-ATA-5, Capacidade de 20,0GB, 7200 RPM's, Tempo de acesso de 11ms, Taxa de tranferencia de 66,6MB/seg, Taxa de transf. Interna de 444MB/seg, Buffer de 2MB, 1"de Altura, 3 anos de garantia	IBM	Dinheiro Cartão	618

Figura 16 - *Site* A



Categoria	Produto	Descricao	Fabricante	Valor
Componentes	Disco rígido	Disco IBM familia Deskstar 75GXP, Interface Ultra-ATA-5, Capacidade de 20,0GB, 7200 RPM's, Tempo de acesso de 11ms, Taxa de tranferencia de 66,6MB/seg, Taxa de transf. Interna de 444MB/seg, Buffer de 2MB, 1"de Altura, 3 anos de garantia	IBM	618
Componentes	Disco rígido	Disco SCSI, Disco rigido Pluggable, Disco rigido padrao Ultra3, Servidores: vide tabela de compatibilidade, Taxa de transferencia de ate 160MB/s, 15.000 RPM, Drive de 1" ou 2.6cm, Capacidade de 18,2GB, Garantia de 1 ano on-site	Compaq	1980
Componentes	Disco rígido	Disco SCSI, Familia Cheetah 18XL, Unidade de disco de 18,4GB Ultra2 160 SCSI 68 LVD pinos, Taxa de transferencia de 284Mbits/seg, Tempo de acesso de 5.2ms, 10.000 rpm, 1,0" de altura, 1.200.000-hour MTBF, 5 anos de garantia	Seagate	1480
Hardware	Computador	Pentium III 500Mhz, 64MB RAM, HD 10GB, CD ROM 36x, FAX/MODEM 56K, 1 ano de garantia	Acer	2699
Hardware	Computador	Pentium III 800Mhz, 128MB RAM, HD 10GB, CD ROM 40x, FAX/MODEM 56K, 2 ano de garantia	IBM	4499
Hardware	Computador	Pentium II 400Mhz, 64MB RAM, HD 10GB, CD ROM 40x, Placa de Video 4Mb AGP, FAX/MODEM 56K, 1 ano de garantia	COMPAQ	2000
Hardware	Impressora	Impressora HP 690 colorida 7 páginas por minuto garantia de 2 anos	HP	699
Hardware	Impressora	Bubble Jet (jato de tinta), Velocidade: 5 ppm em preto (1500 caracteres por página), 2 ppm em colorido (15% de cobertura), Resolução: 720 x 360 ppp, Tamanho do Papel A4, Carta (Letter), Ofício (Legal), Executive, A5, B5 Sutema Operacional: Windows 95, 98, NT 4.0 e 2000, Mac OS 8.1 ou superior, Conectividade: Interface paralela (compatível com ECP IEEE 1284) e USB, Alimentação: AC 120V 60Hz, Consumo Em funcionamento: 30 watts, Nível de Ruído 45 dB (A) ISO 9296, Garantia (balcão) de 3 anos	ELGIN	399
		Notebook Compaq presario 1200 AMD K6 II 450 Mhz, 32MB RAM, HD		

Figura 17 - *Site* B

6.2. Criação do servlet para aplicação de busca na internet

A aplicação tem como objetivo demonstrar que a utilização da XML na construção de catálogos eletrônicos permite a criação de um mecanismo de busca eficiente para Internet. Tal aplicação simulará uma empresa especializada em prover informações aos usuários da Internet sobre produtos de informática - cotação do melhor preço e forma de pagamento. A figura 18 exibe a interface da aplicação, a qual recebeu o nome de Busca Fácil.

Aspectos principais da aplicação a destacar:

- Os produtos a serem pesquisados serão denominados tanto por seu nome técnico, como por nomes conhecidos popularmente.
- Os fornecedores participantes do sistema de busca deverão, obrigatoriamente, possuir um catálogo em XML seguindo os padrões definidos pela prestadora de serviços (no caso, a padronização abordada no item 6.2).
- A aplicação realiza a pesquisa em três fornecedores, podendo ser realizada em quantos forem cadastrados no sistema, desde que sigam a padronização.

Para a realização da pesquisa, o cliente informará a categoria, o tipo, a descrição do produto, e dará um clique no botão “efetuar consulta”. A solicitação será enviada ao *Servlet*, o qual se encarregará de procurar o item solicitado e retornará o resultado ao *browser* do cliente.

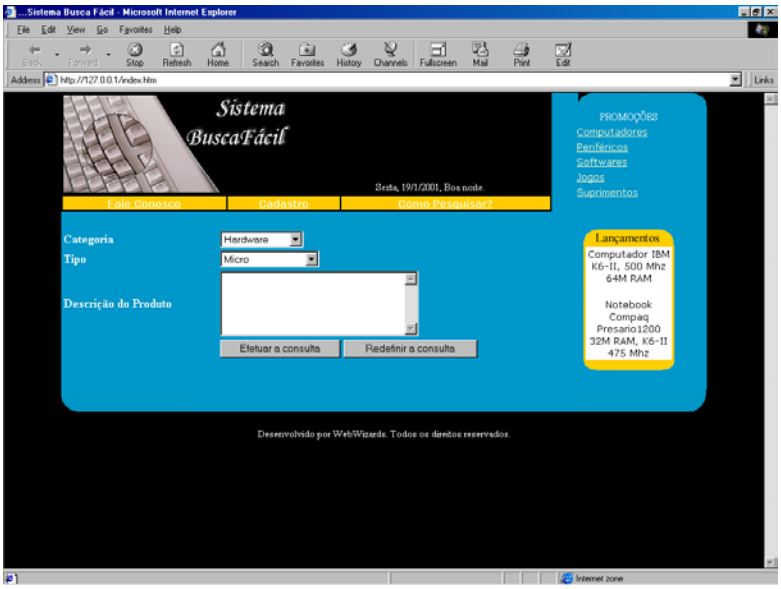


Figura 18 – Interface da Aplicação Busca Fácil

A seguir abordar-se-ão alguns aspectos importantes à pesquisa.

O *Servlet* é o programa responsável por receber os elementos solicitados para a pesquisa, executar a pesquisa nos catálogos dos fornecedores e gerar a página HTML de resposta dinamicamente. A página de resposta é dinâmica porque é sempre diferente, ou seja, dependerá da solicitação de informação efetuada pelo usuário.

A pesquisa no catálogo, simples arquivo texto, é efetuada pelo *parser* SAX, o qual recebe as informações de eventos a serem localizados, segundo procedimento descrito no item relacionado ao SAX. A análise do conteúdo das informações carreadas pelo SAX é efetuada pela aplicação, que seleciona os itens conforme os parâmetros definidos pelo usuário, preparando os dados a serem exibidos através do *servlet.*

A escolha pelo SAX deveu-se à questão de eficiência na manipulação de grande volume de dados, os quais levariam muito tempo para serem lidos e para ser criada uma árvore do documento com o DOM, e o processamento simultâneo de muitos documentos pelo servidor não seria tão eficiente.

O mecanismo de busca se apóia em 4 classes:

- Internet;
- Buscador;
- Produto;

- Parametros.

A classe Internet é estendida da classe HttpServlet, a qual é responsável pelo provimento do serviço de processamento no servidor, atuando: no recebimento dos parâmetros de pesquisa; na criação das instâncias da classe buscador para cada *site* cadastrado, conforme apresentado no quadro 3; na determinação do menor preço entre os produtos selecionados nos sites cadastrados, como mostrado nas linhas de código do quadro 4; no retorno, ao *browser* do cliente, do resultado da busca em formato HTML com as informações sobre descrição, preço, quantidade disponível, forma de pagamento dos produtos pesquisados em cada um dos fornecedores, adicionado a um resumo com o melhor preço encontrado, ou seja, destaque ao preço mínimo encontrado para o produto pesquisado. A figura 19 mostra o resulta de uma busca.

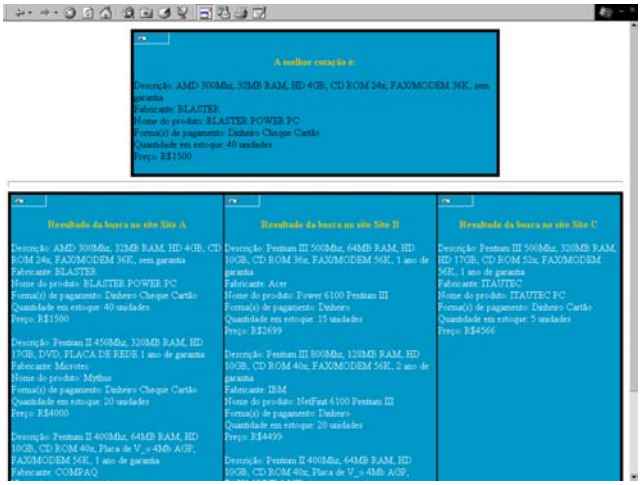


Figura 19 – Resultado da Busca

```
/* criação das Instâncias da Classe Buscador para os sites cadastrados */
buscador b = new buscador(p);
buscador a = new buscador(p);
buscador c = new buscador(p);

a.parseURI("http://127.0.0.1/one.xml");
b.parseURI("http://127.0.0.1/two.xml");
c.parseURI("http://127.0.0.1/three.xml");
```

Quadro 3 – Instância da Classe Buscador

```
/* Retorna o menor preço para o produto selecionado */

  if(a.retornaMenorPreco() > -1 )
menoresPrecos[0] = (produto)a.comResultados.get(a.retornaMenorPreco());
  if(b.retornaMenorPreco() > -1 )
menoresPrecos[1] = (produto)b.comResultados.get(b.retornaMenorPreco());
  if(c.retornaMenorPreco() > -1)
menoresPrecos[2] = (produto)c.comResultados.get(c.retornaMenorPreco());
}catch(Exception e){/*out.println("Erro");*/}


  java.util.Arrays.sort(menoresPrecos, new java.util.Comparator()
  {
    public int compare(Object oA,Object oB)
    {
      if(((produto)oA).valor.equalsIgnoreCase("@@@@@") &&
((produto)oB).valor.equalsIgnoreCase("@@@@@"))
      return 1;
      else if(((produto)oA).valor.equalsIgnoreCase("@@@@@"))
      return 1;
      else if(((produto)oB).valor.equalsIgnoreCase("@@@@@"))
      return - 1;
      if(((produto)oA).retornaValor() < ((produto)oB).retornaValor())
return -1;
else if(((produto)oA).retornaValor() > ((produto)oB).retornaValor())
return 1;
else
return 0;}}) ;
```

Quadro 4 - Determinação do menor preço para o produto

A classe Buscador é estendida da Classe Handlerbase. Tem como atribuição o recebimento dos parâmetros da Classe Internet; recuperação dos dados do site, através da leitura do documento XML; retorno dos elementos que satisfazem aos parâmetros; armazenamento das informações num vetor com a indicação do menor preço dentre os produtos selecionados. O quadro 5 demonstra a criação da instância da SAXParser para iniciar a leitura do documento XML.

```
/* Criação da instância da Classe SAXParser */
  public void parseURI(String uri)
  {
    SAXParser parser = new SAXParser(); // instância
    parser.setDocumentHandler(this);
    parser.setErrorHandler(this);
    try
    {
      parser.parse(uri);
    }
    catch (Exception e)
    {
      System.err.println(e);
    }
  }
```

Quadro 5 – Criação da Instância da Classe SAXParser

A Classe Produto possue a definição da estrutura do documento XML, em substituição ao DTD, facilitando a manipulação dos dados através de métodos implementados em Java. Tem como função retornar ao *servlet* a formatação, em HTML, dos elementos componente do produto.

A Classe Parametros é responsável pela declaração dos parâmetros possíveis para a realização da busca;

7. Conclusões e considerações finais

A aquisição de um bem ou serviço deixará de ser um ato social e se tornará cômoda para o consumidor, haja vista o cliente não precisar mais sair de casa ou do escritório para comprar um produto; ou seja, não terá de enfrentar trânsito, lojas cheias e o desconforto de não encontrar o produto desejado após algumas horas de procura e caminhada num *shopping* ou centro comercial.

O comércio eletrônico é a mais nova forma de aquisição de bens e serviços, possui um grande potencial de crescimento para os próximos anos, em especial para as médias e pequenas empresas, as quais passam a desfrutar de uma tecnologia passível de alcançar consumidores em qualquer parte do planeta. Tal tecnologia permite integrar informações de diversas empresas com redução de custos e ampliação do mercado consumidor.

Os catálogos eletrônicos são considerados, pela maioria das empresas, apenas como uma forma de divulgação de produtos e serviços, em substituição ao modelo tradicional, na forma de papel.

Esta pesquisa demonstrou possuírem, os catálogos eletrônicos, uma nova funcionalidade: proporcionar a integração entre empresas e parceiros de negócio. A interoperabilidade presente nos catálogos admite a participação de qualquer tipo e tamanho de empresa em negociações afins, pois não há a necessidade da determinação de sistemas proprietários. Todas essas possibilidades geradas pelos catálogos estão baseadas na linguagem XML, consolidando-se como uma poderosa ferramenta para os parceiros de negócio.

Para uma efetiva integração em larga escala e um maior compartilhamento de informações, é necessária a elaboração de vários padrões, dentre os quais: criação e divulgação da informação; estruturação de documentos para os diversos ramos de atividade; protocolos mais adequados às novas necessidades de transmissão de dados.

Todo documento pode ser elaborado pelas estruturas hierárquicas da XML, proporcionando aos mecanismos de busca encontrarem a informação de maneira

eficiente e permitindo a evolução dos atuais serviços de pesquisa disponíveis na Internet, já que permite o surgimento de novas empresas especializadas em vender um novo produto: a informação localizada.

A presente dissertação mostra ser possível utilizar o par XML e JAVA como produção alternativa para empresas de recursos financeiros escassos, pois são flexíveis quanto à utilização para o desenvolvimento de ferramentas de trabalho para internet e intranet, proporcionando a integração com parceiros de negócios. Como exemplo, o Tribunal de Justiça de Rondônia, o qual está utilizando XML e JAVA para montar seu Banco de Dados Textual sobre Jurisprudência, Legislação.

A XML é considerada como a plataforma padrão para transações de *e-business* por ter a versatilidade de ser utilizada em qualquer aplicação empresarial: a própria XML não é a resposta, é padrão base no qual podem ser construídas as respostas. Com a ausência de regras, tanto o poder da XML como o perigo estão presentes no *e-business*.

A XML ainda será bastante usada como um repositório comum de metadados, facilitando a troca de informações na Internet, bem como nas transações B2B, com economia de tempo e dinheiro dos parceiros de negócios, nas milhares transações.

É importante ressaltar que os parceiros, na internet e intranet, devem possuir uma linguagem comum para trocar informação estruturada entre os seus sistemas de computador. Sendo assim, a padronização é o ponto chave para o sucesso dessa parceria. A XML é o agente possibilitador, pois, através de *Tags,* definem-se significado e estrutura da informação, permitindo às ferramentas de computador usarem as informação diretamente.

7.1. Trabalhos futuros

Como trabalhos futuros sugere-se:

- Adoção de mecanismos de segurança para os catálogos;
- Análise ergonômica e das tarefas realizadas pelos usuários na utilização do Sistema Busca Fácil;
- Criação de ferramentas de conversão de XML para os outros formatos;

- Criação de instrumentos e metodologias confiáveis para medição de audiência da Internet;
- Integração EDI-XML.

8. Referências bibliográficas

[ADAMS, 1995] ADAMS, Douglas. The Hitchhiker's Guide to the Galaxy. 1.ed. New York: Ballantine Books, 1995.

[AUSWEGER & FRISCH, 2000] AUSWEGER, Matthias; FRISCH, Walter. E-Commerce , Bussines to Bussines: InformationsSysteme "Science on the Web". Disponível em: <http://www.cosy.sbg.ac.at/~maus/e-commerce/frisch#index>. Acesso em: setembro 2000.

[CAP GEMINI, 1999] – CAP GEMINI, Eletronic Commerce – Business issues. Disponível em: <http://www.capgemini.com/ecom/>. Acesso em: 25 setembro 1999.

[COLLA, 1998] COLLA, Ernesto Coutinho. Servlets, Java do lado do servidor. Developers' Magazine, Rio de Janeiro, ano 2, n.23, jul.1998.

[COMMERCENET, 1997] Catalogs for the Digital Marketplace, CommerceNet Research Report 97–03, Palo Alto, March 1997.

[DAHL & LESNICK, 1996] DAHL, A & LESNICK L. Internet Commerce New Riders Publishing, 1996.

[DANIEL, WILSON & PEPPARD, 1999] DANIEL, Elizabeth; WILSON, Hugh; PEPPARD, Joe. Effective Strategies for Electronic Commerce Review of Current Writings and Research. Cranfield-UK: Information Systems Research Centre - Cranfield School of Management, 1999.

[DC, 1998] RFC 2413, Dublin Core Metadata for Resource Discovery ftp://ftp.isi.edu/in– notes/rfc2413.txt>.

[DERTOUZOUS, 1997] DERTOUZOS, Michael. What Will Be: how the new world of information will change our live. 1.ed. Harper USA,1997.338 p.

[DUARTE, 1998] DUARTE, Keesje.Cartões de crédito e o futuro do comércio eletrônico. Artigos – CT & MA, Brasília,07,1998. Disponível em: <http://www.radiobras.gov.br/abrn/c&t/artigos/artigo.htm>. Acesso em: maio 2000.

[FRANCO, 1999] FRANCO, Carlos Alberto Di. Internet e Pornografia. Mirandum, São Paulo, ano1, n. 3, set-dez 1997. Disponível em: <http://www.hottopos.com/mirand3/internet.htm>. Acesso em: 28 outubro 1999.

[GANANÇA, 1998] – GANANÇA, Alexandre Sobino. A Internet e a questão dos direitos autorais. Disponível em: <http://www.persocom.com.br/brasilia/sobrino.htm>. Acesso em: 15 dezembro 1999.

[GOMES, 2000] GOMES, Olavo José Anchieschi. Segurança Total. São Paulo:Makron Books, 2000. 276p.

[KALAKOTA, 1995] Electronic commerce, Lecture Series at Simon School of Business, University of Rochester, 1995.

[KALAKOTA, 1996] KALAKOTA, Ravi. WHINSTON, Andrew. Electronic Commerce : A Manager's Guide.1. ed..California: Addison–Wesley Publishing Company, 1996.

[KALAKOTA & WHINSTON,1996] ___. Frontiers of Electronic Commerce, Addison–Wesley, 1996

[KALAKOTA, 1997] KALAKOTA, Ravi. Readings in Electronic Commerce. Addison– Wesley, 1997.

[KELLER, 1997] ARTHUR M. Keller. Smart Catalogs and Virtual Catalogs, in Readings in Electronic Commerce, Ravi Kalakota and Andrew Whinston, eds., Addison–Wesley, 199., Disponível em : <http://www–db.stanford.edu/pub/keller/#Virtual–Cat>. Acessado em: outubro 2000.

[LIGHT, 1998] LIGHT, Richard. Iniciando em XML. Tradução de Neilande de Moraes, Revisão de Roberto Gabriel Labrada. São Paulo: Makron Books, 1998.

[LOEB, 1998] LOEB, L. Secure Electronic Transactions Artech House, Inc, 1998.

[MARCHAL, 2000] MARCHAL, Benôit. XML conceitos e aplicações. Tradução de Daniel Vieira. São Paulo: Berkeley Brasil, 2000.

[MEGGINSON, 1998] MEGGINSON, David. SAX: The Simple API for XML. Disponível em: <http://megginson.com/SAX/SAX1/index.html>. Acesso em: dezembro 2000.

[MORGADO, 1998] MORGADO, Maurício Gerbaudo. Internet como mídia de marketing direto: estudo exploratório. São paulo, 1998. Dissertação (Mestrado em Administração) - Faculdade de Economia e Administração, Universidade de São Paulo.

[ROBIE, 1998] ROBIE, Johnathan. The Design of XQL. Disponível em: <http://www.texcel.no/whitepapers/xql–design.html>.Acesso em: outubro 2000.

[ROCHA, 1996] ROCHA, H .Como criar a sua Home Page HTML.Rio de Janeiro: infobook, 1996.

[SEARCHTOOLS, 2000] SEARCHTOOLS CONSULTING AND AVI RAPPOPORT. Disponível em: <http://www.searchtools.com/relat/xml.html>.Acesso em:outubro 2000.

[SEGEV, WAN & BEAM, 1995] SEGEV, D. WAN, AND C. BEAM. Designing Electronic Catalogs for Business, CITM Working Paper CITM–WP–1005. October, 1995.

[SOUZA, CATARINO , SANTOS , 1997]. SOUZA, Terezinha Batista de.CATARINO, Maria Elisabete. SANTOS, Paulo César dos. METADADOS: CATALOGANDO DADOS NA INTERNET. Revista Transinformação, Campinas, v. 9, n.2, mai-ago 1997. Disponível em: <http://www.puccamp.br/~biblio/>. Acesso em: novembro 2000.

[TANENBAUM, 1996] TANENBAUM, Andrews S.. Criptografia, Redes de Computadores. Rio de Janeiro: Ed. Campus, 1996.

[TEXCEL, 1998] Texcel International Partners with Microsoft and webMethods to Submit XQL Proposal to the World Wide Web Consortium. Disponível em: <http://www.texcel.no/pr18.htm>. Acesso em: outubro 2000.

[TIDWELL, 1999]. TIDWELL, Doug. Tutorial: XML Programming in Java. Disponível em: <http://www.ibm.com/developer/xml>. Acesso em: setembro 1999.

[VENETIANER, 1996] VENETIANER, T.. HTML: desmistificando a linguagem da Internet. São Paulo:Makrons Books, 1996. 249p.

[XML, 2000] Extensible Markup Language (XML) 1.0 (Second edition). World Wide Web Consortion Recommendation 06- October-2000. Disponível em: <http://www.w3c.org/TR/REC-xml-20001006>. Acesso em: 15 novembro 2000.

Anexos

Os Anexos estão divididos em quatro partes, as quais são básicas para o desenvolvimento da Aplicação Busca Fácil.
Anexo A – DTD, estrutura padronizada do catálogo para possibilitar máxima eficiência do mecanismo de busca.
Anexo B – Folha de estilo XSL.
Anexo C – Catálogo em XML respeitando o modelo do DTD.
Anexo D – Codigo fonte da Aplicação, ou seja, o código Java do servlet.

Anexo A – DTD

```
<!-- edited with XML Spy v3.0.7 (http://www.xmlspy.com) by Joao Carlos Nobre
(estudante) -->
<!-- Criado por Joao Carlos Nobre, UFSC, 2001. -->
<!-- DTDTOOLBARSLIST="" -->
<!ELEMENT COMERCIOELETRONICO ANY>
<!ATTLIST COMERCIOELETRONICO NOME CDATA #IMPLIED>
<!ELEMENT PRODUTO (CATEGORIA, TIPO, FABRICANTE, NOME,
DESCRICAO, VALOR, QUANTIDADE, PAGAMENTO)>
<!ELEMENT CATEGORIA (#PCDATA)>
<!ELEMENT TIPO (#PCDATA)>
<!ELEMENT FABRICANTE (#PCDATA)>
<!ELEMENT NOME (#PCDATA)>
<!ELEMENT DESCRICAO (#PCDATA)>
<!ELEMENT VALOR (#PCDATA)>
<!ELEMENT QUANTIDADE (#PCDATA)>
<!ELEMENT PAGAMENTO (#PCDATA)>
```

Anexo B – Folha de estilo XSL.

```
<?xml version='1.0'?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
  <html>
          <body>
            <table border="2" bgcolor="#AABBAA">
              <tr>
                        <th>Categoria</th>
                        <th>Produto</th>
                        <th>Descricao</th>
                        <th>Fabricante</th>
                        <th>Forma de Pagamento</th>
                        <th>Valor</th>
              </tr>
              <xsl:for-each select="COMERCIOELETRONICO/PRODUTO" order-by =
        "+ TIPO">
              <tr>
                        <td><xsl:value-of select="CATEGORIA"/></td>
                        <td><xsl:value-of select="TIPO"/></td>
                        <td><xsl:value-of select="DESCRICAO"/></td>
                        <td><xsl:value-of select="FABRICANTE"/></td>
                        <td><xsl:value-of select="PAGAMENTO"/></td>
                        <td><xsl:value-of select="VALOR"/></td>
              </tr>
              </xsl:for-each>
            </table>
          </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```

Anexo C – Catálogo XML

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- edited with XML Spy v3.0.7 (http://www.xmlspy.com) by Joao Carlos Nobre
(estudante) -->
<!-- Criado por Joao Carlos Nobre, UFSC, 2001.-->
<COMERCIOELETRONICO NOME="ONE">
	<PRODUTO>
		<CATEGORIA>Hardware</CATEGORIA>
		<TIPO>Computador</TIPO>
		<FABRICANTE>BLASTER</FABRICANTE>
		<NOME>BLASTER POWER PC</NOME>
		<DESCRICAO>AMD 300Mhz, 32MB RAM, HD 4GB, CD ROM 24x,
FAX/MODEM 36K, sem garantia</DESCRICAO>
		<VALOR>1500</VALOR>
		<QUANTIDADE>40 unidades</QUANTIDADE>
		<PAGAMENTO>Dinheiro Cheque Cartão</PAGAMENTO>
	</PRODUTO>
	<PRODUTO>
		<CATEGORIA>Hardware</CATEGORIA>
		<TIPO>Computador</TIPO>
		<FABRICANTE>Microtec</FABRICANTE>
		<NOME>Mythus</NOME>
		<DESCRICAO>Pentium II 450Mhz, 320MB RAM, HD 17GB, DVD,
PLACA DE REDE 1 ano de garantia</DESCRICAO>
		<VALOR>4000</VALOR>
		<QUANTIDADE>20 unidades</QUANTIDADE>
		<PAGAMENTO>Dinheiro Cheque Cartão</PAGAMENTO>
	</PRODUTO>
	<PRODUTO>
		<CATEGORIA>Hardware</CATEGORIA>
		<TIPO>Computador</TIPO>
		<FABRICANTE>COMPAQ</FABRICANTE>
		<NOME>Armada</NOME>
		<DESCRICAO>Pentium II 400Mhz, 64MB RAM, HD 10GB, CD ROM
40x, Placa de Vídeo 4Mb AGP, FAX/MODEM 56K, 1 ano de
garantia</DESCRICAO>
		<VALOR>2000</VALOR>
		<QUANTIDADE>20 unidades</QUANTIDADE>
		<PAGAMENTO>Dinheiro</PAGAMENTO>
	</PRODUTO>
	<PRODUTO>
		<CATEGORIA>Hardware</CATEGORIA>
		<TIPO>Notebook</TIPO>
		<FABRICANTE>SONY</FABRICANTE>
		<NOME>VAIO</NOME>
```

```
                <DESCRICAO>Notebook SONY VAIO K6 II 450 Mhz, 32MB RAM,
HD 5GB, CD ROM 24X</DESCRICAO>
                <VALOR>5699</VALOR>
                <QUANTIDADE>15 unidades</QUANTIDADE>
                <PAGAMENTO>Dinheiro Cheque</PAGAMENTO>
        </PRODUTO>
</COMERCIOELETRONICO>
```

Anexo D - Código Fonte da Aplicação Busca Fácil

INTERNET

```
package buscafacil;

/**
* Title:        Busca fácil
* Description:
* Copyright:    Copyright (c) 2001
* Company:
* @author JCN
* @version 1.0
*/
import java.util.*;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class internet extends HttpServlet
{
public void service(HttpServletRequest request,
HttpServletResponse response)throws ServletException, IOException
{
produto menoresPrecos[] = {new produto(),new produto(),new produto()};
response.setContentType("text/html");
PrintWriter out = response.getWriter();
parametros p = new parametros();
p.categoria = request.getParameter("categoria") ;
p.tipo = request.getParameter("tipo") ;
p.descricao = request.getParameter("descricao") ;

buscador a = new buscador(p);
buscador b = new buscador(p);
buscador c = new buscador(p);

a.parseURI("http://127.0.0.1/one.xml");
b.parseURI("http://127.0.0.1/two.xml");
c.parseURI("http://127.0.0.1/three.xml");
try{
if(a.retornaMenorPreco() > -1 )
menoresPrecos[0] = (produto)a.comResultados.get(a.retornaMenorPreco());
if(b.retornaMenorPreco() > -1 )
menoresPrecos[1] = (produto)b.comResultados.get(b.retornaMenorPreco());
if(c.retornaMenorPreco() > -1)
menoresPrecos[2] = (produto)c.comResultados.get(c.retornaMenorPreco());
}catch(Exception e}
```

```
java.util.Arrays.sort(menoresPrecos, new java.util.Comparator()
{
public int compare(Object oA,Object oB)
{
if(((produto)oA).valor.equalsIgnoreCase("@@@@@") &&
((produto)oB).valor.equalsIgnoreCase("@@@@@"))
return 1;
else if(((produto)oA).valor.equalsIgnoreCase("@@@@@"))
return 1;
else if(((produto)oB).valor.equalsIgnoreCase("@@@@@"))
return - 1;
if(((produto)oA).retornaValor() < ((produto)oB).retornaValor())
return -1;
else if(((produto)oA).retornaValor() > ((produto)oB).retornaValor())
return 1;
else
return 0;}}) ;

/*
out.println(menoresPrecos[0].valor+"<br>");
out.println(menoresPrecos[1].valor+"<br>");
out.println(menoresPrecos[2].valor+"<br>");
*/


if(!menoresPrecos[0].valor.equalsIgnoreCase("@@@@@"))
{
out.println("<table width = \"60%\" border=\"4\" bgcolor=\"#0099CC\" vspace=\"0\"
hspace=\"0\" cellpadding=\"0\" cellspacing=\"0\" bordercolor=\"#000000\"
align=\"center\"><tr>");
out.println("<td valign=\"top\"><b><font color=\"#FFCC00\">");
out.println("<p align=\"left\"><img
src=\"http://127.0.0.1/images/topoEsqResultPeq.gif\" width=\"57\"
height=\"19\"></p>");
out.println("<center>A melhor cotação é:</font></b></center><br>");
out.println(menoresPrecos[0].retornaFormatado() );
out.println("</td></tr><tr><td>"+"<a href =
"+"http://127.0.0.1/index.htm><center><font color=\"#FFCC00\"> Nova Pesquisa
</font></center></a>"+"</td></tr></table>");

}



out.println("<hr size=\"10\">");
```

```
out.println("<html><head><title>Resultado da busca</title></head><body
bgcolor=\"#000000\">");

out.println("<table border=\"4\" bgcolor=\"#0099CC\" vspace=\"0\" hspace=\"0\"
cellpadding=\"0\" cellspacing=\"0\" bordercolor=\"#000000\"><tr>");
//out.println("<table width=\"100%\" border=\"0\" bgcolor=\"#0099CC\"><tr>");
////tabela
processa(a,out,"Site <a href = "+"http://127.0.0.1/one.html> A </a>" );
processa(b,out,"Site <a href = "+"http://127.0.0.1/two.xml> B </a>");
processa(c,out,"Site <a href = "+"http://127.0.0.1/three.html> C </a>");
out.println("</tr></table><br>"); ////fecha tabela
out.println("</body></html>");
}

public void processa(buscador x, PrintWriter out, String site){
if(x.retornaMenorPreco()>-1){

out.println("<td valign=\"top\">"); ////coluna
out.println("<p align=\"left\"><img
src=\"http://127.0.0.1/images/topoEsqResultPeq.gif\" width=\"57\"
height=\"19\"></p>");
out.println("<center><b><font color=\"#FFCC00\">Resultado da busca no site "+site+"
</font></b></center><br><hr size=\"10\">");

for(int i=0; i<x.comResultados.size(); i++){
out.println("<font color=\"#FFFFFF\">");
out.println(((produto)x.comResultados.elementAt(i)).retornaFormatado());
out.println("</font>");
out.println("<hr size=\"10\">");
}
out.println("</td>"); ////fecha coluna
}else
{
out.println("<td valign=\"top\"><b><font color=\"#FFCC00\">");
out.println("<p align=\"left\"><img
src=\"http://127.0.0.1/images/topoEsqResultPeq.gif\" width=\"57\"
height=\"19\"></p>");
out.println("<center>A pesquisa no site "+site+" não retornou nenhum
resultado</center><br><br>");
out.println("</font></b></td>");
}
}
}
```

BUSCADOR

```
package buscafacil;
import java.io.*;
import java.util.*;
```

```
import org.xml.sax.*;
import org.xml.sax.helpers.ParserFactory;
import com.ibm.xml.parsers.SAXParser;

public class buscador
extends HandlerBase
{
int elementos;
String temp = "";
Vector comResultados = new Vector();
produto prodTemp = new produto();
parametros par;

public buscador(parametros param){
par = param;
}

public void parseURI(String uri)
{
SAXParser parser = new SAXParser();
parser.setDocumentHandler(this);
parser.setErrorHandler(this);
try
{
parser.parse(uri);
}
catch (Exception e)
{
System.err.println(e);
}
}


/** Characters. */
public void characters(char ch[], int start, int length)
{
temp = new String(ch,start,length);
} // characters(char[],int,int);

/** End element. */

public void endElement(String name)
{
if(name.equalsIgnoreCase("categoria") )
prodTemp.categoria  =  temp;
if(name.equalsIgnoreCase("tipo") )
prodTemp.tipo = temp;
if(name.equalsIgnoreCase("fabricante") )
```

```
prodTemp.fabricante = temp;
if(name.equalsIgnoreCase("nome") )
prodTemp.nome = temp;
if(name.equalsIgnoreCase("descricao") )
prodTemp.descricao = temp;
if(name.equalsIgnoreCase("valor") )
prodTemp.valor = temp;
if(name.equalsIgnoreCase("pagamento") )
prodTemp.pagamento = temp;
if(name.equalsIgnoreCase("quantidade") )
prodTemp.quantidade  = temp;
if(name.equalsIgnoreCase("produto"))
checa();
} // endElement(String)


public void checa(){
if(
(prodTemp.categoria.toUpperCase().indexOf(par.categoria.toUpperCase()) > -1) &&
(prodTemp.tipo.toUpperCase().indexOf(par.tipo.toUpperCase()) > -1) &&
/*(prodTemp.descricao.toUpperCase().indexOf(par.descricao.toUpperCase()) > -1)*/
checaDescricao()
)
comResultados.add(prodTemp);

prodTemp = null;
prodTemp = new produto();
//prodTemp.zera();

}

public int retornaMenorPreco(){
if(comResultados.size() > 0){
int premiado = 0;
float tempPreco = Float.parseFloat( ((produto)comResultados.get(0)).valor );
//  System.out.println(tempPreco);

//  System.out.println(comResultados.size());
for(int i = 0; i<comResultados.size() ; i++)
{
if(tempPreco > Float.parseFloat( ((produto)comResultados.get(i)).valor ))
{
tempPreco = Float.parseFloat( ((produto)comResultados.get(i)).valor );
premiado = i;
}
}
return premiado;
}
```

```
else
return -1;
}

public boolean checaDescricao(){
StringTokenizer aosPedacos = new StringTokenizer(par.descricao," ");
String varStrTemp = "";
int conComCampos = aosPedacos.countTokens() ;
int tempCampos = 0;
int indexInicio;
while(aosPedacos.hasMoreElements())
{
if((prodTemp.descricao.toUpperCase().indexOf(((String)aosPedacos.nextElement()).to
UpperCase())) != -1)
tempCampos++;
}
if(conComCampos == tempCampos)
return true;
else
return false;
}

/** End document. */
public void endDocument()
{
//  System.out.println(this.retornaMenorPreco());
} // endDocument()



/** Fatal error. */
public void fatalError(SAXParseException ex)
throws SAXException
{
throw ex;
}

/** Main program entry point. */
}
```

PRODUTO

```
package buscafacil;

/**
* Title:        Busca fácil
* Description:
* Copyright:    Copyright (c) 2001
* Company:
```

```
* @author JCN
* @version 1.0
*/

public class produto {
public String categoria = "@@@@@";
public String tipo = "@@@@@";
public String fabricante = "@@@@@";
public String nome = "@@@@@";
public String descricao = "@@@@@";
public String quantidade = "@@@@@";
public String valor = "@@@@@";
public String pagamento = "@@@@@";
public void zera(){
categoria = tipo = fabricante = nome = descricao = quantidade = valor = pagamento =
"0";
}
public String retornaFormatado(){
return "Descrição: "+descricao +"<br>" + "Fabricante: " + fabricante  +"<br>"+
"Nome do produto: "+nome+"<br>"+
"Forma(s) de pagamento: " + pagamento  + "<br>"+
"Quantidade em estoque: " + quantidade  + "<br>" +
"Preço: R$" + valor  + "<br><br>";
}

public float retornaValor(){
try{
return Float.parseFloat(valor) ;
}catch(Exception e){ return 999999999;}
}
}
```

PARAMETROS

```
package buscafacil;

/**
* Title:        Busca fácil
* Description:
* Copyright:    Copyright (c) 2001
* Company:
* @author JCN
* @version 1.0
*/

public class parametros {
public String categoria;
public String tipo;
public String fabricante;
```

```
public String nome;
public String descricao;
public String quantidade;
public String valor;
public String pagamento;
public parametros() {
}
}
```